URBAN 3D SCENE UNDERSTANDING FROM IMAGES

A Dissertation
Presented to
The Academic Faculty

By

Abhijit Kundu

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in Computer Science at
School of Interactive Computing

Georgia Institute of Technology

May 2018

URBAN 3D SCENE UNDERSTANDING FROM IMAGES

Approved by:

Dr. James M. Rehg, Advisor
School of Interactive Computing
Georgia Institute of Technology

Dr. Frank Dellaert, co-advisor
School of Interactive Computing
Georgia Institute of Technology

Dr. James Hays
School of Interactive Computing
Georgia Institute of Technology

Dr. Vladlen Koltun
Principal Researcher
Intel Labs

Dr. Marc Polleyfeys
Director of Science
Microsoft HoloLens

Date Approved: January 9th, 2018

What I cannot create, I do not understand.

Richard Feynman

For my father.

ACKNOWLEDGEMENTS

I should like to begin by extending a thanks to my PhD advisors, Professor Jim Rehg and Professor Frank Dellaert. Jim has always encouraged my independent thought and has allowed me an amount of autonomy in my research many students can only dream of. I have come out of every meeting with Jim with a renewed source of enthusiasm and motivation. And to Professor Frank Dellaert, I owe a debt of gratitude for teaching me so much of multi-view geometry. The guidance of both these men has made this work possible, and they have been great mentors and role models. I'd like to especially thank them for their endless patience, to which I was often not entitled, but appreciated very much. And for their confidence in me, unwavering through many stumbles and falls, I am thankful.

Let me also express my gratitude to the rest of my committee: Professor Vladlen Koltun, Professor Marc Pollefeys, and Professor James Hays. Thank you all for lending me your time and your ear as a part of my committee. I should also thank Vladlen in particular for his mentorship during an incredible internship at Intel Labs.

My sincere thanks also goes to the many more friends and colleagues, too numerous to name in this brief interlude, whom I've had the pleasure of working with.

Finally to my family: Mother, Brother, thank you. My work has separated us, physically. But never did you allow it to draw me from your hearts and your thoughts. Your love and support has been palpable throughout this endeavor, and has oft been the spur that urges me to persevere through the many obstacles one encounters in doing his PhD.

And the man to whom this thesis is dedicated, my father. Father, it pains me that you cannot share in the celebrations of this accomplishment. It was you who moulded my perspective of the world, who showed me the joy in building something of your own. In the lonely mornings that followed the many arduous, sleepless nights of the past six years, I thought of you most. May you endure through this work of mine, for this creation of *your* creation, is for you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

A fundamental goal of computer vision is to construct structured 3D representations of complex dynamic scenes from 2D image sequences. Motivated by applications in autonomous driving, our focus is on complex urban street scenes. The primary challenge is to construct a consistent, semantically rich, and complete representation of the scene consisting of both dynamic objects, such as vehicles and pedestrians, and static background structures, such as sidewalks, streets, and buildings. In this thesis, we explore three different scene understanding frameworks with increasing richness in representation. The presented frameworks reason jointly about the scene structure and their semantic labels, along with 3D orientation and position of object instances over time. We also demonstrate seamless integration of different constraints and prior knowledge into our model and an effective fusion of measurements from multiple images in a video into a final representation of the scene. We evaluate these scene understanding frameworks on challenging real-world datasets of complex urban scenes.

The task of scene understanding is to obtain a compact representation of the scene that makes subsequent tasks easier. In this thesis, we argue for a rich 3D scene representation. Given an image, our system estimates the amodal 3D shape, 3D pose, and semantic category of all object instances in the scene. The output of our system can be directly used for tasks like path-planning, augmented-reality, or to accurately predict an object's 3D location in the future. Another major benefit of scene understanding with a rich 3D scene model is that traditional 2D scene representations (*e.g.* semantic segmentation, instance segmentation, and 2D depth maps) may be obtained for free. They can be generated by simply rendering the output 3D scene model. In contrast, traditional scene understanding methods are designed specifically for only a single task, like instance segmentation or depth map.

A 2D image is a formed by a complex function of several attributes including lighting, shape, and surface properties of objects in the scene. Unlike 2D projection, an instance-

level 3D understanding provides a disentangled representation of the scene. But how do we invert the image formation process to obtain the 3D scene model? One classical approach to solving inverse problems in vision is *analysis-by-synthesis*. It consists of using a model that describes the observed data generation process (*synthesis*), which is then used to estimate the parameters of the model that generated the particular observed data (*analysis*). *Analysis-by-synthesis* with a 3D scene model is like **"solving vision as *inverse-graphics*"**. *Synthesis* describes the process of generating image content from the 3D scene model in the style of *computer graphics*. Vision is then like doing *analysis* by searching the best 3D scene configuration to explain the observed image. While a conceptually elegant and mature idea, *inverse-graphics* has, as of yet, only been successful for a very limited number of problems. This is due to the fact that typical 3D scene representation is very high-dimensional, and analysis then becomes a difficult search problem over these large, high-dimensional scene variables.

Recently, there has been a re-emergence of *inverse-graphics* approach in which an efficient, discriminative bottom-up method like a convolutional network is used to reduce the search space. However, most of these approaches are still restricted to simple scenes often containing only one object. We present an *inverse-graphics* approach which is capable of handling complex real-world 3D scenes. Our approach uses a deep convolutional network to map image regions to 3D representations of all object instances in an image. We exploit class-specific shape priors by learning a low dimensional shape-space from CAD model collections. We present novel representations of shape and pose, that strive towards better 3D equivariance and generalization. In order to exploit a rich amount of supervisory signals in the form of 2D annotations, like segmentation, we also present a differentiable Render-and-Compare loss that allows 3D shape and pose to be learned with 2D supervision.

1

INTRODUCTION

3D understanding of complex scenes from 2D images is a long standing goal of computer vision and such capabilities open up a lot of applications which have not previously been possible. We humans are marvelous in obtaining a structured representation of any scene such as spatial scene-layout, re-organization of the scene with its constituent objects, support of each object, *etc*. We also see the complete extent of the scene, even parts of the scene or objects which are occluded. For example, even when part of the scene directly below a car is not visible, we infer that it is a part of the road. This kind of structured and complete *perception* of the scene is essential for many real-world applications, such as autonomous driving, planning, automated game level creation, cartography, and other robotics applications.

Our objective is to build a 3D scene model from images which is structured, semantically rich, compact, and complete in extent. In this thesis, we present three scene understanding frameworks with increasing richness in representation. Each framework solves different aspects of the problem of scene understanding from 2D images. The presented frameworks reasons jointly about the scene structure, their semantic labels along with 3D orientation and position of object instances over time. We also demonstrate seamless integration of different constraints and prior knowledge into our model and an effective fusion of measurements from multiple images in a video into a final prediction.

> *"Perception is our best guess as to what is in the world, given our current sensory evidence and our prior experience."*
>
> – Helmholtz

1.1 Scene Understanding

The scene understanding term has been used in computer vision to broadly describe high-level understanding of the image content. It can be described as *extracting a compact representation of the image that is easily accessible to subsequent tasks*.

1.1.1 Why we need scene understanding?

Since scene understanding is only a part of the full processing pipeline for some task, one valid question is that, "Do we need scene understanding?". For example, in autonomous driving, one can train a neural network directly from images to control in an end-to-end learning framework [1], completely bypassing perception.

Though there are counter arguments that such end-to-end learning will not be possible [2] due to lack of labelled data, such systems lack important qualities like introspection. Such systems lack an abstract knowledge of perception, that can be reused for other tasks. They have to be re-trained for every other robot with different control mechanisms like drones, or even different models of car. Such approaches are highly tied to the training data and only demonstrate minor generalization capability. According to Tenenbaum *et al*. [3], *"human minds make inferences that go far beyond the data available and generalizes to many more tasks with much less training data"*.

1.1.2 What constitutes a good scene understanding representation?

The task of scene understanding is to obtain a compact representation of the scene that makes subsequent tasks easier and more feasible. Therefore, there is no universal "good" scene representation, only good scene representation with respect to a set of objectives, and task in hand. However, there is still a set of common best practices and ideals that we want in our "good" scene understanding representations.

Representation choice is a critical component when designing systems for scene under-

standing. Judea Pearl [4] argues that humans superseded other species due to *"their ability to sketch and store a representation of their environment, interrogate that representation, distort it by mental acts of imagination and finally answer What if? kind of questions."*. We discuss some of the desirable properties of scene representation below:

- We want our representations to be *generic*, so that they can be reused for multiple tasks. As an added benefit, such representations can then get training signals while performing all those tasks.
- Capability to *predict* future state is an important tenant of any intelligent system. It is also important for applications like self-driving cars, where we want to predict the behavior of other cars and pedestrian for safe operation.
- A *disentangled* and *interpretable* representation is also important. A disentangled representation makes it easy to understand and introspect causes of failure. It also factorizes the representation, and allows for direct supervision when only a subset of the representation parameters has ground-truth labels.
- We need representations that forces the learning agent to *study harder* and avoid shortcuts. A good representation also make it is *easy to express priors and constraints* that we know about the world.

In the next section, we argue for semantically rich 3D scene representation, which capture the properties discussed above and has several advantages over traditional scene understanding representations for applications in our 3D world like self-driving cars.

1.1.3 3D scene understanding

Till recent years, traditional scene understanding was only concerned about assigning semantic labels to pixels in just a single image or finding 2D bounding boxes around objects of interest. However such 2D representations are often inefficient for tasks like path planning, 3D spatial reasoning.

In this thesis, we argue for semantically rich 3D scene representation. Our method outputs 3D shape and pose along with semantic category of each object instance in the scene. Since the representation captures instance-level semantic information, we can have both class-specific and instance-specific semantic priors. The detailed 3D scene representation makes it possible to reason about interactions between scene elements in 3D space. This allows for a proper handling of interactions like occlusion, collision and 3D support.

Constraints like spatial smoothness, motion smoothness, shape constancy are better expressed with an explicit 3D representation. However, most of the observations and training data are available in 2D image domain. 3D representation with differentiable rendering capability offers the best of both worlds, as the 3D representation can be readily rendered and compared with 2D observations.

3D scene understanding provides a *disentangled* representation of the scene: 3D shape and pose of object instances over time. This *disentangled* 3D representation makes it possible to enforce specific priors on object shape or motion. The output from our system can be directly used for tasks like navigation, path-planning, accurately predicting a object's 3D location in the future.

Estimating the full 3D shape and pose of an object is a much harder task than simple semantic classification tasks commonly practiced in computer vision. For instance, most learning algorithms trained to classify object categories, often pickup misleading correlations like *"green grass"* when classifying *cows*. Such systems will fail to generalize to an image of a *"cow in a beach"*. By asking the learning agent to predict the complete 3D shape and pose parameters of an object, forces the learning agent to *study harder*, avoid shortcuts and generalize better.

Another major benefit of doing a scene understanding with a rich 3D scene model, is that traditional 2D scene representations like semantic segmentation, instance segmentation, object detection and tracking are all available for free. They can be generated by simply rendering the output 3D scene model. In contrast, traditional scene understand-

ing methods are designed specifically for only a single task like semantic segmentation or object detection.

1.2 Inverse-Graphics: An *Analysis-by-Synthesis* approach for Vision

A 2D image is a formed by a complex function of several attributes like cameras, lighting, structure and surface properties of objects of the scene. The task of 3D scene understanding then involves inverting the complex image formation process to obtain the 3D scene model.

Analysis-by-synthesis is an old idea for solving inverse problems. It comprises of using a model that describes the observed data generation process (*synthesis*), which is then used to find the parameters of the model that generated a particular observed data (*analysis*). The idea of *analysis-by-synthesis* can be rooted back to Helmhotz's 1867 work on unconscious inference [5]. Another influential work in this domain is the work of *Helmholtz machine* by Hinton *et al*. [6, 7]. The *analysis-by-synthesis* approach for computer vision also has a long history [8, 9, 10].

In this thesis, we propose an *analysis-by-synthesis* framework for 3D scene understanding. We present an object-centric 3D scene model and a novel, differentiable, top-down analysis framework. *Analysis-by-synthesis* with a 3D scene model is like *"solving vision as inverse graphics"*. *Synthesis* describes the process of forming image from the 3D scene model in the style of *computer graphics*. Vision is then like doing *analysis* by searching the best 3D scene configuration to explain the observed image.

But it must be emphasized that having a 3D scene model is not necessarily required for analysis-by-synthesis. A vast majority of vision applications in segmentation, stereo, optical-flow uses a 2D image-space model. In Chapter 2, we use a 2D image space model for video, and we show how traditional 2D smoothness prior can be quite bad (more so for video). As a solution, we present a better way to incorporate smoothness prior that have the characteristics of a true 3D smoothness prior. However in Chapter 3 and Chapter 4, we use 3D scene model. This also allows us to model prior constraints like smoothness and

collision much more naturally.

1.2.1 Main challenges of Inverse-Graphics

In all generative models, including *analysis-by-synthesis*, there is always a trade-off between model fidelity and hardness of inference. A richer model can always explain the data well, but that comes at the cost of an increase in parameter space of the model, making it harder to infer.

While it is important to have a scene model which is rich enough to describe arbitrary complex images, it leads to a large high-dimensional search space for scene variables. So analysis then becomes a difficult search problem over this large, high-dimensional scene latent variables. Secondly, the posterior distribution of the scene variables is usually very multi-modal and could easily lead an optimization algorithm to get stuck in local minima.

The analysis-by-synthesis approach to vision offers an elegant and conceptually simple account for the remarkable capability of human vision. But a true generative analysis-by-synthesis approach is typically considered too slow to explain perception in the brain. This is the direct consequence of the large, high-dimensional search space. One solution of cutting down the search space, is to have a fast bottom-up process that directly predicts some of the scene variables, which is then used to initialize or narrow down the search for complete scene parameters. Related work on combining with top-down analysis with bottom-up methods include [10, 11, 12] and more recently in [13, 14, 15].

1.2.2 Why we need to revisit Inverse-Graphics

Even though *inverse-graphics* offers an intuitive and elegant solution, it has only been successful for very limited problems. It is much less popular than relatively fast discriminative bottom-up approaches with data-intensive training. But lately, there is a renewed interest in *inverse-graphics* [16, 17, 18, 13, 19, 20, 15, 21, 22], coinciding with success of deep learning. In these new approaches to *inverse-graphics*, a discriminative bottom-up method

like a convolutional neural network is used to cut down on the large search space. Ability of deep convolutional neural networks for provides a very promising solution.

Availability of large-scale 3D CAD model repositories [23, 24] recently is also significant. This can be useful to learn class-specific 3D shape priors. Availability of large-scale annotated urban image datasets with with 3D information [25, 26, 27, 28, 29, 30] in the last couple of years is also a significant enabling factor for learning and evaluating 3D scene understanding algorithms.

However most of inverse graphics approaches including the recent approaches are still restricted to toy problems, or only work on limited scenes typically with only one object in the scene. In Chapter 4 of this thesis we present an fast *inverse-graphics* approach which is capable of handling complex real-world 3D scenes.

1.3 Thesis Organization

The remainder of this thesis proposal document is organized in three different chapters with increasing richness of representation: image space representation in Chapter 2, static 3D representation in Chapter 3, and finally we propose a object-centric (instance-level) 3D representation in Chapter 4.

Each chapter solves a slightly different aspect of the overall problem of ***"scene understanding from videos"***. In all of the chapters, we have a top-down structured prediction model that jointly reasons about all the scene elements. Additionally, this top-down structured prediction benefits from a bottom-up discriminative classifier. However, only in Chapter 4, we present the fully end-to-end trained *inverse-graphics* approach, discussed previously in this chapter. In Chapter 4, we also use an instance-level representation. Both Chapter 2 and Chapter 3 provide a semantic understanding of the scene but lack the concept of object instances. We now present a brief summary of each chapter.

Chapter 2: We present a structured-prediction framework for semantic video segmentation, that jointly assigns semantic labels to all pixels in the video. We employ a fully connected spatio-temporal CRF for this purpose. We show that a standard temporal extension of 2D image-space regularization can be very detrimental for videos, as they do not capture temporal correspondence information. We solve for a low-dimensional feature space embedding that captures correspondence, and can be effectively used for regularization. The framework can be used atop any bottom-up unary classifiers and we show improved semantic segmentation both in accuracy and temporal consistency.

Chapter 3: We do a semantic 3D reconstruction of the static scene from a monocular image stream. We formulate this problem as spatial 3D CRF with several novel factors to account for sparse 3D reconstruction and ambiguities in obtaining the 3D world from 2D images. We use a 3D volumetric representation, where each voxel is globally assigned a semantic label indicating whether it is a free voxel or otherwise its solid semantic object category. We demonstrate both improved 3D reconstruction and semantic segmentation over traditional approach of solving them separately.

Chapter 4: We present a fast *inverse-graphics* framework for instance-level 3D scene understanding. We train a deep convolutional network that learns to map image regions to full 3D shape and pose of all object instances in the image. Our method produces a compact 3D representation of the scene, which can be readily used for applications like autonomous driving. Many traditional 2D vision tasks like instance segmentation and depth-maps can be obtained by simply rendering our output 3D scene model. We exploit class-specific shape priors by learning a low dimensional shape-space from CAD model collections. We present novel representations of shape and pose, that strive towards better 3D equivariance and generalization. In order to exploit rich amount of supervisory signals in form of 2D annotations like segmentation, we propose a differentiable Render-and-Compare loss module that allows 3D shape and pose to be learned with 2D supervision. We evaluate

on challenging real-world datasets of Pascal3D+ and KITTI where our method achieves state-of-the-art results.

1.4 List of Publications

The following publications form the basis for this thesis:

- **Abhijit Kundu**, Yin Li, and James M Rehg. 3D-RCNN: Instance-level 3D Scene Understanding via Render-and-Compare. Under submission for **CVPR**, 2018.
- **Abhijit Kundu**, Vibhav Vineet, and Vladlen Koltun. Feature Space Optimization for Semantic Video Segmentation. In **CVPR**, 2016.
- **Abhijit Kundu**, Yin Li, Frank Dellaert and James M Rehg. Joint Semantic Segmentation and 3D Reconstruction from Monocular Video. In **ECCV**, 2014.
- Florian Hauer, **Abhijit Kundu**, James M Rehg, Panagiotis Tsiotras. Multiscale Perception and Path Planning on Probabilistic Obstacle Maps. In **ICRA**, 2015.
- Vikas Dhiman, **Abhijit Kundu**, Frank Dellaert and Jason J. Corso. Modern MAP inference methods for accurate and fast occupancy grid mapping on higher order factor graphs. In **ICRA**, 2014.

2

SEMANTIC VIDEO SEGMENTATION

In this chapter we present a top-down structured prediction framework for semantic video segmentation. Temporal regularization in video is challenging because both the camera and the scene may be in motion. Thus Euclidean distance in the space-time volume is not a good proxy for correspondence. We optimize the mapping of pixels to a Euclidean feature space so as to minimize distances between corresponding points. Structured prediction is performed by a dense CRF that operates on the optimized features. Experimental results demonstrate that the presented approach increases the accuracy and temporal consistency of semantic video segmentation. This chapter's work has been published as [31].

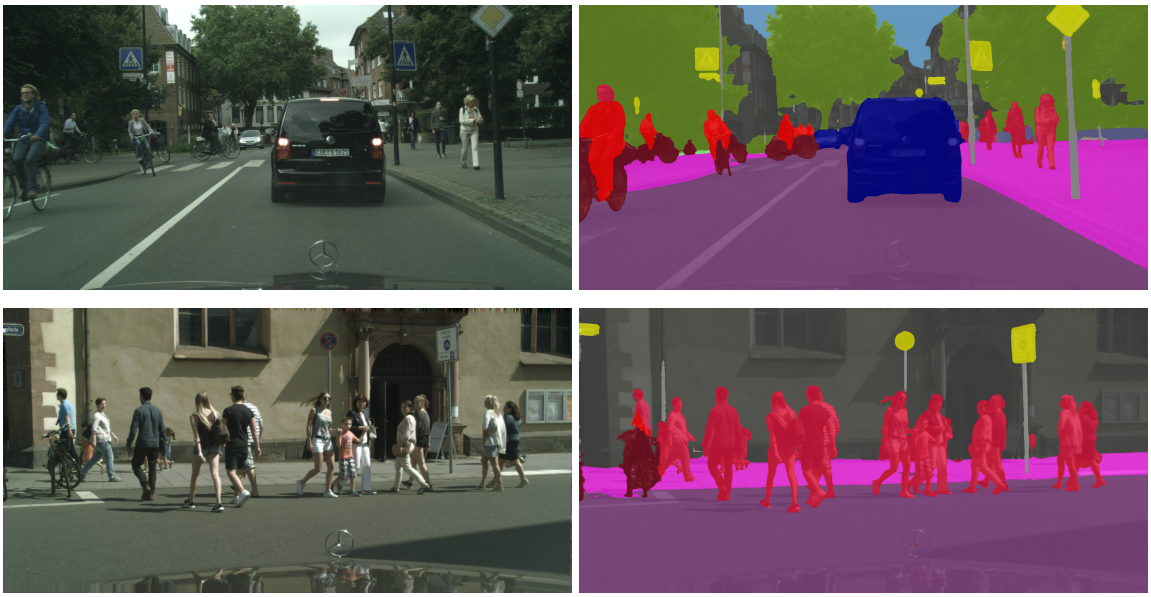

Figure 2.1: Semantic video segmentation on the Cityscapes dataset [27]. Input frame on the left, semantic segmentation computed by our approach on the right.

2.1 Introduction

Structured prediction has become a standard means of achieving maximal accuracy in semantic segmentation. In structured prediction, all pixels are labeled jointly and labeling coherence is explicitly enforced. This alleviates the noise and inconsistency that can arise when pixels are classified independently. In particular, the fully-connected CRF [32, 33] – also known as the dense CRF – often yields significant improvements in semantic segmentation accuracy. For example, after the recent breakthrough of Long et al. [34], who developed a new model for semantic image segmentation, an application of the dense CRF over the new model yielded substantial accuracy gains [35, 36, 37].

The natural form of input for vision systems that operate in the physical world is video. For this reason, we consider semantic segmentation of video sequences, rather than individual images. In a typical video sequence, each frame depicts a different view of the scene. Thus structured prediction can be used not only for spatial regularization within individual frames but also for temporal consistency across frames. In this paper, we address the challenges brought up by such spatio-temporal regularization.

Long-range temporal regularization in video is complicated by the fact that both the camera and the scene may be in motion. In particular, camera motion can induce significant optical flow across the visual field. For example, when the camera rotates, a point in the scene can quickly translate across the image plane. For this reason, simply appending the time dimension to the feature space used for regularization can lead to incorrect associations and cause misprediction in the presence of significant camera and object motion. The underlying problem is that Euclidean distance in the space-time video volume is not a good proxy for correspondence.

Our solution is to optimize the feature space used by the dense CRF so that distances between features associated with corresponding points in the scene are minimized. The dense CRF operates on an embedding of the pixels into a Euclidean feature space [32]. The

Euclidean norm in this space is used to define a continuous measure of correspondence. All pairs of pixels are connected and all pairs of pixels are regularized. In our setting, the regularization is performed over a fully-connected graph over the video volume. The strength of the connection between a pair of pixels is a function of their distance in the feature space. Our approach optimizes the feature space embedding such that Euclidean distance in feature space is a more accurate measure of correspondence in the underlying scene.

Specifically, we establish temporal correspondences via optical flow and long-term tracks and optimize the feature space embedding to minimize distances between corresponding points, subject to second-order regularization constraints. We express the embedding objective as a linear least-squares problem and show that feature space optimization can be performed efficiently over high-resolution video volumes. The resulting embedding is used by a fully-connected space-time CRF that performs direct long-range regularization across the video volume, while operating at full resolution and producing sharp pixel-level boundaries.

We evaluate the proposed semantic video segmentation approach through extensive experiments on the CamVid and Cityscapes datasets [38, 27]. Experimental results demonstrate that feature space optimization increases the accuracy of semantic video segmentation. Our approach yields a 66.1% mean IoU on CamVid and a 70.3% mean IoU on the Cityscapes validation set. Both results are the highest reported to date. In addition, the presented approach substantially increases the temporal consistency of the labeling. This is evaluated quantitatively in our experiments and is also evident in the supplementary video. Figure 2.1 shows results produced by the presented approach on two frames from the Cityscapes dataset.

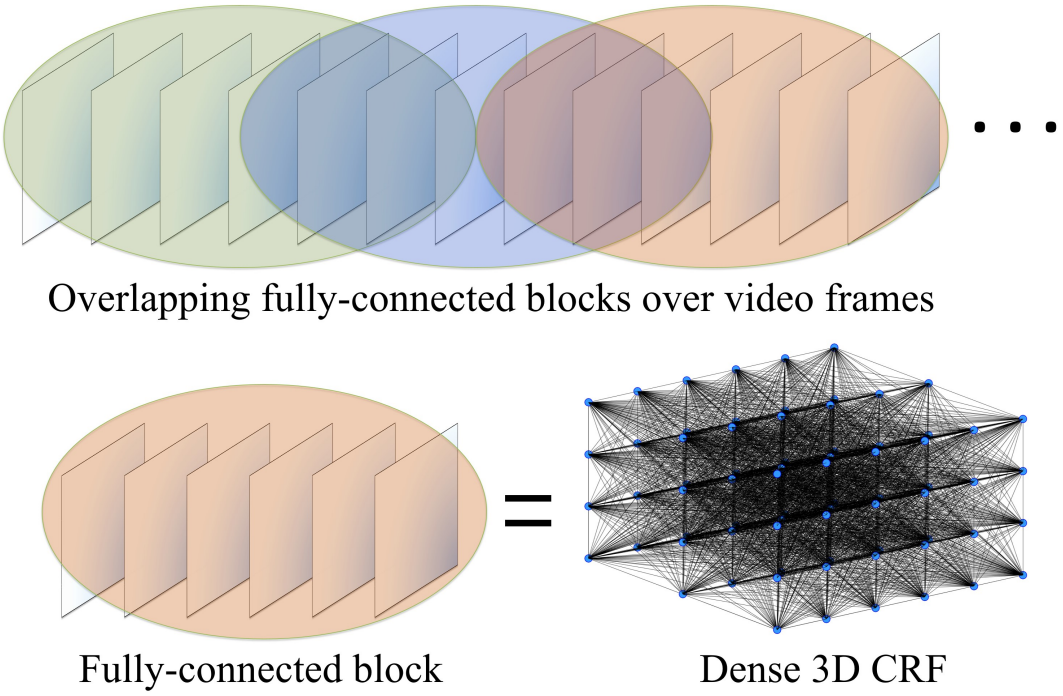

Figure 2.2: The temporal structure of the model. The video is covered by overlapping blocks. A dense CRF is defined over each block and feature space optimization is performed within blocks. Structured prediction is performed over multiple blocks.

2.2 Model

Our model is a set of cliques that cover overlapping blocks in the video volume. We cover the video by overlapping temporal blocks, define a dense CRF over each block, and build in provisions for temporally smooth prediction across block boundaries. The temporal structure of the model is illustrated in Figure 2.2. The block construction is described in § 2.5.

Each pixel in the video is identified by a vector $\mathbf{p} = (b, t, i) \in \mathbb{R}^3$, where b is the block number, t is the frame number within block b, and i is the index of the pixel within the frame. The color of pixel $\mathbf{p}$ is denoted by $\mathbf{I}_\mathbf{p} \in \mathbb{R}^3$ and the coordinates of pixel $\mathbf{p}$ in its frame are denoted by $\bar{\mathbf{s}}_\mathbf{p} \in \mathbb{R}^2$. Let $\mathbf{P}$ be the set of pixels in the video.

Given pixel $\mathbf{p}$, let $X_\mathbf{p}$ be a random variable with the domain $\mathcal{L} = \{l_1, ..., l_L\}$. The states l_i will be referred to as labels. Let $\mathcal{X}$ be a random field over $\mathbf{P}$ and let $\mathbf{x} : \mathbf{P} \rightarrow \mathcal{L}$ be a label assignment. The random field $\mathcal{X}$ is characterized by a Gibbs distribution $P(\mathbf{x}|\mathbf{P})$ and

the corresponding Gibbs energy $E(\mathbf{x}|\mathbf{P})$ associated with each label assignment:

$$\begin{aligned} P(\mathbf{x}|\mathbf{P}) &= \frac{1}{Z(\mathbf{P})} \exp\big(-E(\mathbf{x}|\mathbf{P})\big), \\ E(\mathbf{x}|\mathbf{P}) &= \sum_{\mathbf{p}} \psi^u_{\mathbf{p}}(x_{\mathbf{p}}) + \sum_{(\mathbf{p},\mathbf{q})\in\mathcal{E}} \psi^p_{\mathbf{p},\mathbf{q}}(x_{\mathbf{p}}, x_{\mathbf{q}}). \end{aligned} \tag{2.1}$$

Here $Z(\mathbf{P}) = \sum_{\mathbf{x}} \exp\big(-E(\mathbf{x}|\mathbf{P})\big)$ is the partition function and $\mathcal{E}$ is a neighborhood structure defined on pairs of variables. The neighborhood structure is a union of cliques: each block is covered by a clique, each pixel is covered by two blocks, and each variable is correspondingly covered by two fully-connected subgraphs in the random field. Our goal is to find a label assignment $\mathbf{x}^*$ that minimizes the Gibbs energy.

The unary term $\psi^u_{\mathbf{p}}(x_{\mathbf{p}})$ specifies the cost of assigning label $x_{\mathbf{p}}$ to pixel $\mathbf{p}$. Pairwise terms $\psi^p_{\mathbf{p},\mathbf{q}}(x_{\mathbf{p}}, x_{\mathbf{q}})$ couple pairs of variables and penalize inconsistent labeling. These terms are defined using Gaussian kernels [32]:

$$\psi^p_{\mathbf{p},\mathbf{q}}(x_{\mathbf{p}}, x_{\mathbf{q}}) = \mu(x_{\mathbf{p}}, x_{\mathbf{q}}) \sum_{m=1}^{M} w^m \kappa^m(\mathbf{f}_{\mathbf{p}}, \mathbf{f}_{\mathbf{q}}), \tag{2.2}$$

where $\mu(x_{\mathbf{p}}, x_{\mathbf{q}})$ is a label compatibility term and w^m are the mixture weights. $\mathbf{f}_{\mathbf{p}}$ and $\mathbf{f}_{\mathbf{q}}$ are features associated with $x_{\mathbf{p}}$ and $x_{\mathbf{q}}$, respectively. Each kernel has the following form:

$$\kappa^m(\mathbf{f}_{\mathbf{p}}, \mathbf{f}_{\mathbf{q}}) = \exp\left(-\frac{\|\mathbf{f}_{\mathbf{p}} - \mathbf{f}_{\mathbf{q}}\|^2}{\sigma_m^2}\right). \tag{2.3}$$

Given point $\mathbf{p}$, the feature $\mathbf{f}_{\mathbf{p}} \in \mathbb{R}^D$ is a vector in a D-dimensional feature space. In semantic image segmentation, the canonical feature space is five-dimensional and combines image position and color [32]. A natural feature space for semantic video segmentation is six-dimensional and combines time, color, and position: $\mathbf{f}_{\mathbf{p}} = (t_{\mathbf{p}}, \mathbf{I}_{\mathbf{p}}, \bar{\mathbf{s}}_{\mathbf{p}})$. We will use this feature space as a starting point.

2.3 Feature Space Optimization

Feature-sensitive models of the kind described in § 2.2 have been very successful in semantic image segmentation [32, 37]. However, applying such models to space-time video volumes is not straightforward. A key difficulty is that both the camera and the objects may be in motion and can carry corresponding pixels apart. Thus the natural six-dimensional feature space yields a distance measure that does not appropriately model spatio-temporal correspondence.

A hypothetical solution that would address this issue is to obtain a dense metric 3D reconstruction of the scene through time, associate each pixel with the true 3D position of the corresponding surface element in the environment, and use this 3D position along with time as a feature. This would enforce a coherence assumption on surfaces that are truly proximate in space-time. However, dense monocular 3D reconstruction of dynamic scenes is an open problem. We therefore develop an alternative approach that does not require understanding the three-dimensional layout of the scene.

Our approach involves optimizing a subspace of the feature space to reduce Euclidean distance between corresponding points while adhering to regularization terms that aim to preserve object shapes. Specifically, for all points $\{\mathbf{p}\}$, we optimize position features $\{\mathbf{s_p}\}$. (The time and color dimensions are fixed.) Thus the feature mapping $(t_\mathbf{p}, \mathbf{I_p}, \bar{\mathbf{s}}_\mathbf{p})$ is replaced by $(t_\mathbf{p}, \mathbf{I_p}, \mathbf{s_p})$.

Consider a block b that consists of $T \times N$ points, where T is the number of frames in the block and N is the number of pixels in each frame. The optimization objective is defined as follows:

$$\begin{aligned} \mathbf{s}^* &= \arg\min_{\mathbf{s}} E(\mathbf{s}), \\ E(\mathbf{s}) &= E_u(\mathbf{s}) + \gamma_1 E_s(\mathbf{s}) + \gamma_2 E_t(\mathbf{s}). \end{aligned} \tag{2.4}$$

Here $\mathbf{s}$ are the position features for all pixels in the block and $\mathbf{s}^*$ are the optimal features. The objective $E(\mathbf{s})$ comprises a data term $E_u(\mathbf{s})$, a spatial regularizer $E_s(\mathbf{s})$, and a temporal

regularizer $E_t(\mathbf{s})$. We now explain each of these three terms. We will use $\mathbf{p}$ and (b, t, i) interchangeably to denote a point in the block.

Data term $E(\mathbf{s})$. The data term prevents the feature space embedding from drifting or collapsing under the strength of the regularization terms. The middle frame in the block is used as an anchor. Let $a = \lfloor T/2 \rfloor$ be the frame number of the anchor frame and let $\mathbf{P}^a$ be the set of pixels in frame a. Let $\{\bar{\mathbf{s}}_\mathbf{p} : \mathbf{p} \in \mathbf{P}^a\}$ be the unoptimized natural feature space for $\mathbf{P}^a$. The data term ensures that points in the anchor frame do not drift far from their natural positions:

$$E_u = \sum_{\mathbf{p} \in \mathbf{P}^a} \left(\mathbf{s}_\mathbf{p} - \bar{\mathbf{s}}_\mathbf{p}\right)^2. \tag{2.5}$$

Spatial regularization term $E_s(\mathbf{s})$. The spatial regularizer preserves shapes within color boundaries and detected contours. We use anisotropic second-order regularization over the 4-connected pixel grid [39, 40]:

$$E_s(\mathbf{s}) = \sum_{t=1}^{T} \sum_{i=1}^{N} \left(\mathbf{s}_{(b,t,i)} - \sum_{j \in \mathcal{N}_i} w_{ij} \mathbf{s}_{(b,t,j)} \right)^2. \tag{2.6}$$

Here $\mathcal{N}_i$ is the set of neighbors of point (b, t, i). The weight w_{ij} attenuates the regularization at object boundaries:

$$w_{ij} = \exp\left(-\frac{\|\mathbf{I}_{(b,t,i)} - \mathbf{I}_{(b,t,j)}\|^2}{\sigma_1} \right) \exp\left(-\frac{c_\mathbf{p}^2}{\sigma_2} \right). \tag{2.7}$$

The first factor in (2.7) is based on the color difference between the two pixels and the second factor is based on the contour strength at pixel $\mathbf{p}$. We use structured forests to compute contour strength $c_\mathbf{p}$ [41], such that $c_\mathbf{p} \in [0, 1]$ and $c_\mathbf{p} = 1$ indicates the presence of a boundary.

Temporal regularization term $E_t(\mathbf{s})$. The temporal regularizer pulls corresponding points in different frames to assume similar positions in feature space:

$$E_t(\mathbf{s}) = \sum_{(\mathbf{p},\mathbf{q})\in\mathcal{K}} \left(\mathbf{s_p} - \mathbf{s_q}\right)^2. \tag{2.8}$$

This is the term that minimizes distances between corresponding points. $\mathcal{K}$ is a collection of correspondence pairs $(\mathbf{p}, \mathbf{q})$, where $\mathbf{p}$ and $\mathbf{q}$ are in different frames. Correspondences are established via optical flow and long-term tracks, as described in § 2.5.

Optimization. Objective (2.4) is a large-scale linear least-squares problem with second-order regularization. We optimize the objective using the biconjugate gradient stabilized method [42] with algebraic multigrid preconditioning [43].

2.4 Inference

Efficient inference in the model specified by Equation (2.1) can be performed by an extension of the mean-field inference algorithm introduced by Krähenbühl and Koltun [32]. Note that our model is a collection of overlapping cliques and is thus different from the fully-connected model considered by Krähenbühl and Koltun.

Define a distribution Q that approximates the true distribution P, where similarity between distributions is measured by the KL-divergence. Assume that Q factorizes over the individual variables: $Q(\mathbf{x}) = \prod_{\mathbf{p}} Q_{\mathbf{p}}(x_{\mathbf{p}})$, where $Q_{\mathbf{p}}$ is a distribution over the random vari-

able $X_{\mathbf{p}}$. The mean-field updates have the following form:

$$\begin{aligned} Q_{\mathbf{p}}(l) &= \frac{1}{Z_{\mathbf{p}}} \exp\Big(-\psi^u_{\mathbf{p}}(l) - S_1(l) - S_2(l)\Big), \\ S_1(l) &= -\sum_{l' \in \mathcal{L}} \sum_{\mathbf{q} \in \mathcal{N}^1_{\mathbf{p}}} Q_{\mathbf{q}}(l') \psi^p_{\mathbf{p},\mathbf{q}}(l, l'), \\ S_2(l) &= -\sum_{l' \in \mathcal{L}} \sum_{\mathbf{q} \in \mathcal{N}^2_{\mathbf{p}}} Q_{\mathbf{q}}(l') \psi^p_{\mathbf{p},\mathbf{q}}(l, l'), \\ Z_{\mathbf{p}} &= \sum_{l} \exp\Big(-\psi^u_{\mathbf{p}}(l) - S_1(l) - S_2(l)\Big), \end{aligned} \tag{2.9}$$

where $\mathcal{N}^1_{\mathbf{p}}$ and $\mathcal{N}^2_{\mathbf{p}}$ are sets of neighbors of $\mathbf{p}$ in the two blocks that cover $\mathbf{p}$. The updates can be performed efficiently using Gaussian filtering in feature space [32]. Given the Q distribution at the end of the final iteration, a labeling can be obtained by assigning $x^*_{\mathbf{p}} = \arg\max_l Q_{\mathbf{p}}(l)$.

We now consider what happens when the video volume is too large to fit in memory. We can partition the video into chunks of consecutive blocks, such that inference in each chunk is performed separately. To align the predicted distributions across blocks, we could use a distributed optimization strategy such as dual decomposition [44]. However, the convergence of such schemes can be quite slow. We therefore opt for a simple heuristic that has the added benefit that chunks can be processed in a streaming fashion.

Consider two overlapping blocks b^1 and b^2, such that b^1 is the last block in one chunk and b^2 is the first block in the next chunk. Let Q^1 and Q^2 be the distributions produced by mean-field inference for these blocks in their respective chunks. Let $[t_1, t_2]$ be the overlap region. Let Q^t be the sought-after distribution for frame $t \in [t_1, t_2]$ and let $Q^{1,t}$ and $Q^{2,t}$ be the corresponding slices of Q^1 and Q^2. We transition between chunks via simple linear interpolation:

$$Q^t = Q^{1,t} + \frac{t - t_1}{t_2 - t_1} Q^{2,t}. \tag{2.10}$$

2.5 Implementation

We use two sets of unary potentials in our experiments. The first is the classical TextonBoost classifier of Shotton et al. [45], as implemented by Ladicky et al. [46]. This classifier was used in a number of prior semantic video segmentation systems and enables a fair comparison to prior work. Second, we use a convolutional network based on the work of Yu and Koltun [47], which we refer to as the Dilation unary. This network consists of a front-end prediction module and a context aggregation module. The front-end module is an adaptation of the VGG-16 network based on dilated convolutions [48, 47]. The context module uses dilated convolutions to systematically expand the receptive field and aggregate contextual information [47]. In combination, the two modules form a high-performing convolutional network for dense prediction. In particular, the Dilation network yielded the highest semantic segmentation accuracy among all models evaluated by Cordts et al. [27], without using structured prediction.

In all experiments, we use optical flow computed by LDOF [49]. To evaluate the influence of the input flow, we also conduct a controlled experiment with Discrete Flow [50]. Long-term tracks are computed using the approach of Sundaram et al. [51]. CRF parameters are optimized using grid search on a subset of the validation set.

The decomposition into blocks can be performed using a fixed block size, such as 100 frames. Our implementation uses a different approach that adapts block boundaries to the content of the video. Specifically, we consider long-term tracks [51] and spawn a new block when more than half of the tracks in the frame were not present at the beginning of the block. This increases the internal coherence of each block.

2.6 Experiments

We evaluate the presented approach on two datasets for road scene understanding: the CamVid dataset [38] and the Cityscapes dataset [27]. Both datasets provide video input

along with pixel-level semantic annotations of selected frames.

2.6.1 CamVid dataset

We begin by performing experiments on the CamVid dataset. We use the split of Sturgess et al. [52], which has been adopted in a number of prior works. This split partitions the dataset into 367 training images, 100 validation images, and 233 test images. 11 semantic classes are used.

The primary accuracy measure we use is mean IoU (intersection over union). The IoU score for a particular class is defined as $\frac{\mathrm{TP}}{\mathrm{TP+FP+FN}}$, where TP, FP, and FN is the number of true positives, false positives, and false negatives for this class, respectively [53].

We have also evaluated global pixel accuracy, defined as the total fraction of correctly classified pixels. We have ascertained that our approach outperforms the prior work in terms of pixel accuracy. However, this measure is severely biased in favor of large classes, such as "sky" and "road", and discounts small but important classes such as "pole" or "sign". We therefore do not report it and discourage other researchers from using it.

In addition to accuracy evaluation on frames that have ground-truth label maps, we have also evaluated the temporal consistency of the labeling produced by each technique. To this end, we have defined a consistency measure in terms of long-term tracks [51]. A track is said to be consistently labeled if all pixels along the track are assigned the same label. The consistency of a labeling is defined to be the fraction of tracks that are consistently labeled. Note that perfect consistency can be achieved trivially at the expense of accuracy: all pixels in all frames in the video can simply be assigned the same label. However, a combination of high accuracy and high consistency is not easy to achieve and we have found that high consistency does correspond to qualitative stability.

Ablation study. We first perform a controlled study to isolate the effect of feature space optimization on labeling accuracy. The results of this experiment are provided in Table

2.1. We use the TextonBoost unary [45]. Applying a dense 2D CRF within each frame independently improves both mean IoU and consistency. Applying a dense 3D CRF over the video volume improves both metrics further. Performing feature space optimization as proposed in this paper improves both metrics further still.

	mean IoU	Consistency
TextonBoost unary	47.43	60.88
Dense 2D CRF	51.08	74.37
Dense 3D CRF	53.08	81.68
Our approach	**55.23**	**87.33**

Table 2.1: Ablation study with TextonBoost unaries. Spatio-temporal resularization over the video volume increases both accuracy and consistency. Feature space optimization outperforms the baselines.

	Building	Tree	Sky	Car	Sign	Road	Pedestrian	Fence	Pole	Sidewalk	Bicyclist	mean IoU	Consistency
Without ConvNet													
ALE [46]	73.4	70.2	91.1	64.24	24.4	**91.1**	29.1	31	13.6	72.4	**28.6**	53.59	72.2
SuperParsing [54]	70.4	54.8	83.5	43.3	25.4	83.4	11.6	18.3	5.2	57.4	8.9	42.03	**88.8**
Tripathi et al. [55]	74.2	67.9	91	**66.5**	23.6	90.7	26.2	28.5	16.3	71.9	28.2	53.18	76.8
Liu and He [56]	66.8	66.6	90.1	62.9	21.4	85.8	28	17.8	8.3	63.5	8.5	47.2	77.6
TextonBoost+FSO	**74.4**	**71.8**	**91.6**	64.9	**27.7**	91.0	**33.8**	**34.1**	**16.8**	**73.9**	27.6	**55.2**	87.3
With ConvNet													
SegNet [57]	68.7	52	87	58.5	13.4	86.2	25.3	17.9	16.0	60.5	24.8	46.4	62.5
Dilation [47]	82.6	76.2	89.9	84.0	46.9	92.2	56.3	35.8	**23.4**	75.3	55.5	65.29	79.0
Dilation+FSO	**84.0**	**77.2**	**91.3**	**85.6**	**49.9**	**92.5**	**59.1**	**37.6**	16.9	**76.0**	**57.2**	**66.12**	**88.3**

Table 2.2: Quantitative results on the CamVid dataset [38]. This table reports per-class IoU, mean IoU, and temporal consistency. *Top:* comparison to prior work that did not use convolutional networks. Using the classical TextonBoost classifier [45], our approach outperforms the prior work. *Bottom:* comparison to prior work that used convolutional networks and evaluated on the CamVid dataset. Using the Dilation network [47], our approach (FSO) yields the highest accuracy reported on the CamVid dataset to date.

Comparison to prior work. We now compare the presented approach against state-of-the-art methods for semantic video segmentation. The first set of baseline methods – SuperParsing [54] and the method of Liu and He [58] – perform semantic video segmentation at the supervoxel level. The second set of methods – Tripathi et al. [55] and Miksik et al. [59] – operate at the pixel level. Tripathi et al. define a dense 3D CRF in the space-time

volume, but do not optimize the feature space. Miksik et al. enforce temporal smoothness by other means. Finally, we compare to SegNet, a recent convolutional network that has been evaluated on CamVid [57].

Quantitative results are provided in Table 2.2. (Quantitative comparison against Miksik et al. [59] is provided separately in supplementary material, since Miksik et al. only provided the results of their approach on a subset of the CamVid test set.) Using the classical TextonBoost unary, our approach achieves an accuracy gain of 8 percentage points over the recent method of Liu and He [56] and an improvement of 2 percentage points over Tripathi et al. [55].

The Dilation network outperforms SegNet by 19 percentage points. Feature space optimization and structured prediction yield a further accuracy gain and a 9 percentage point boost in consistency over the Dilation unary. To assess the sensitivity of feature space optimization to the input optical flow, we evaluated the results of feature space optimization when the input flow fields are computed by LDOF [49] and Discrete Flow [50]. However, performance of the approach is virtually identical in the both these two conditions.

Qualitative results are provided in Figure 2.3 and in the supplementary video.

2.6.2 Cityscapes dataset

Cityscapes is a new dataset for scene understanding in urban environments [27]. The dataset contains 2975 training images, 500 validation images, and 1525 test images. 19 semantic classes are used. We report results on the validation set. Results on the test set will be provided in supplementary material.

The results are reported in Table 2.3. We compare to the recent Adelaide model, a comprehensive system that integrates convolutional networks and conditional random fields [36]. The Dilation network yields slightly higher accuracy than the Adelaide model. Using the Dilation unary, our approach yields a further gain in accuracy and an improvement of more than 6 percentage points in consistency.

	mean IoU	Consistency
Adelaide [36]	68.6	-
Dilation unary [47]	68.65	88.14
Dilation + Our approach	**70.30**	**94.71**

Table 2.3: Quantitative results on the Cityscapes validation set.

2.7 Conclusion

We proposed feature space optimization for spatio-temporal regularization. The key observation is that naive regularization over the video volume does not take camera and object motion into account. To support efficient long-range temporal regularization, we optimize the positions of points in the space so that distances between corresponding points are minimized. Applying a dense random field over this optimized feature space yields state-of-the-art semantic video segmentation accuracy.

The presented approach can directly benefit from more accurate optical flow and more stable and temporally extended point trajectories. We encourage further development of these basic building blocks [60, 61]. More broadly, the presented feature space optimization formulation has significant limitations and more flexible approaches should be explored.

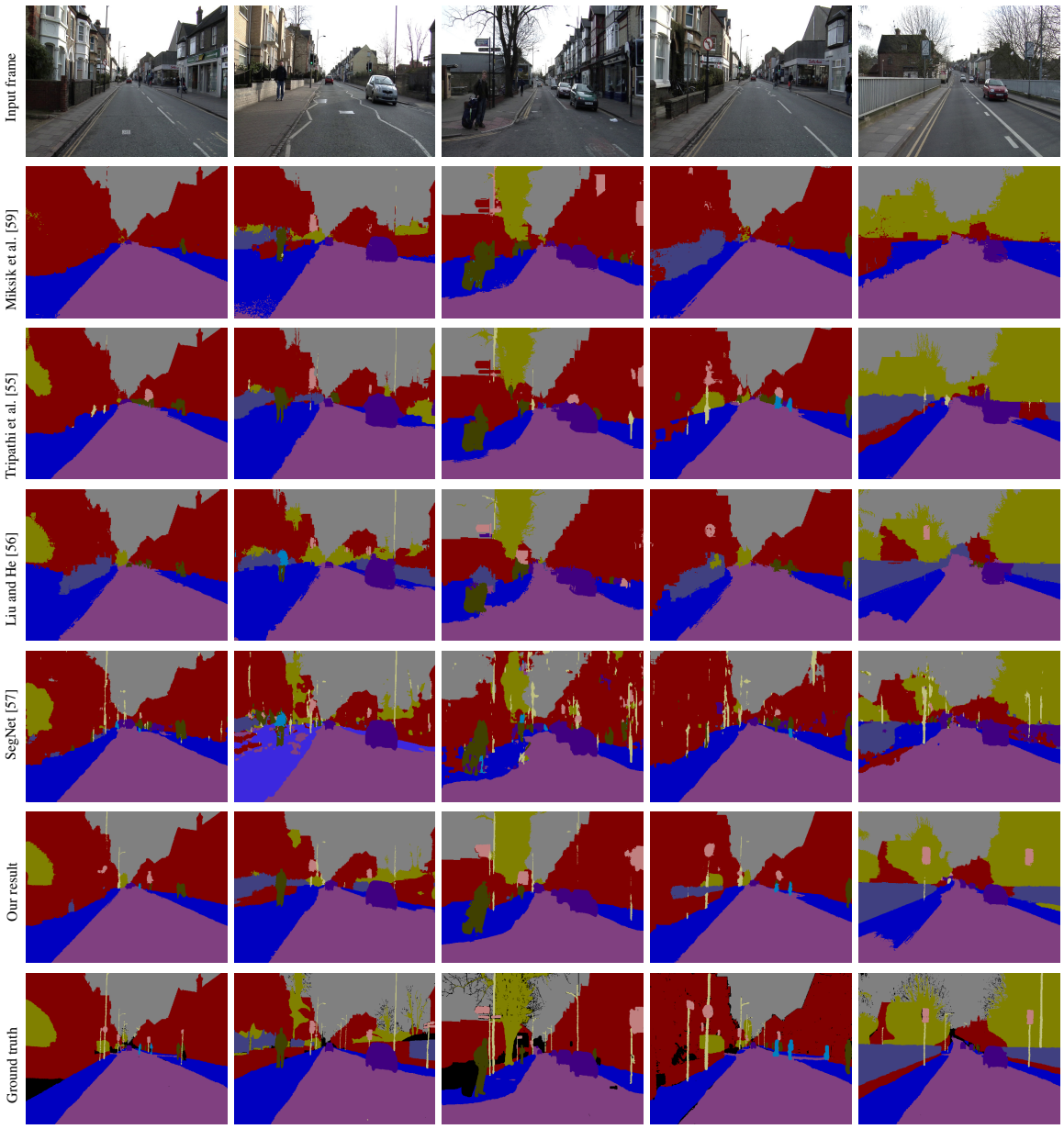

Figure 2.3: Qualitative results on the CamVid dataset [38]. From top to bottom: input frame, Miksik et al. [59], Tripathi et al. [55], Liu and He [56], SegNet [57], semantic segmentation produced by the presented approach, and ground truth.

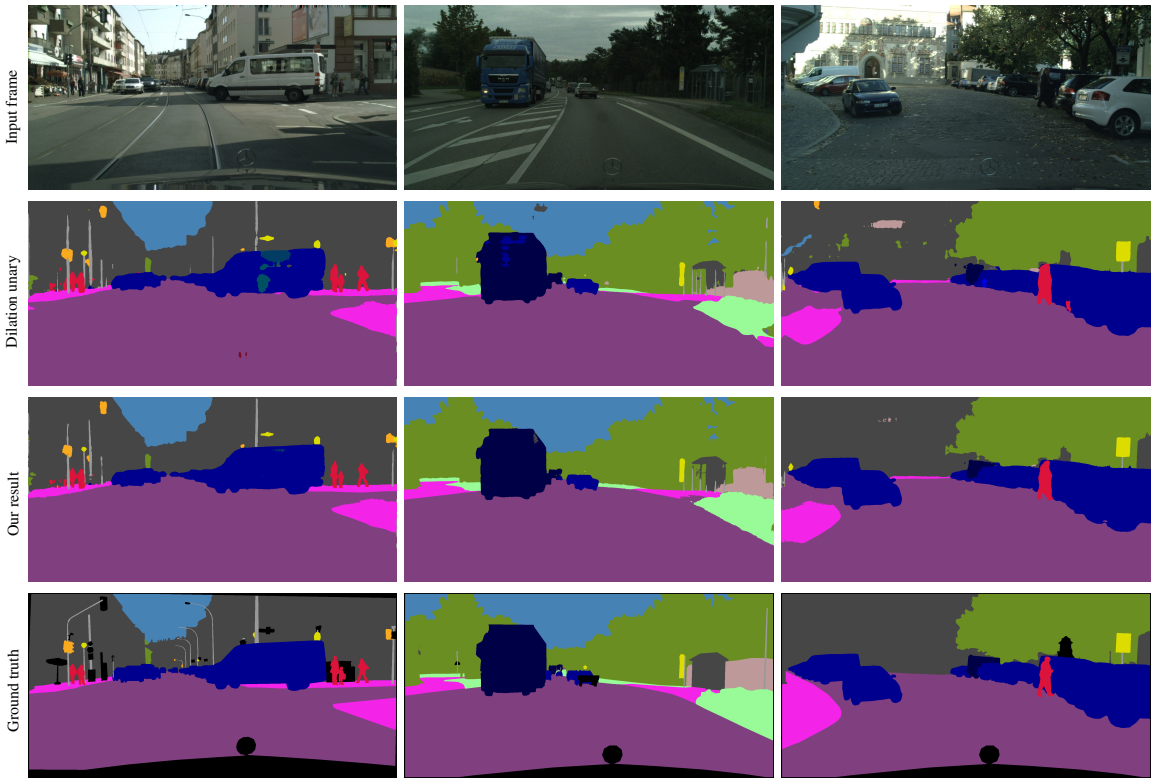

Figure 2.4: Qualitative results on the Cityscapes dataset [27]. From top to bottom: input frame, output from the Dilation unary [47], semantic segmentation produced by the presented approach, and ground truth.

3

SEMANTIC 3D RECONSTRUCTION

In this chapter, we present an approach for joint inference of 3D static scene structure and its semantic labeling. Given a monocular image stream, our framework produces a 3D volumetric semantic + occupancy map, which is much more useful than a series of 2D semantic label images (like in Chapter 2) or a sparse point cloud produced by traditional semantic segmentation and SfM pipelines respectively. We derive a top-down CRF model defined in the 3D space, that jointly infers the semantic category and occupancy for each voxel. Such a joint inference in the 3D CRF paves the way for more informed priors and constraints, which is otherwise not possible if solved separately in their traditional frameworks. We make use of class specific semantic cues that constrain the 3D structure in areas, where multiview constraints are weak. Our model comprises of higher order factors, which helps when the depth is unobservable. We also make use of class specific semantic cues to reduce either the degree of such higher order factors, or to approximately model them with unaries if possible. We demonstrate improved 3D structure and temporally consistent semantic segmentation for difficult, large scale, forward moving monocular image sequences. This chapter is primarily based on our previous work that appeared in [62].

3.1 Introduction

To successfully navigate and perceive the 3D world, a robot needs to infer both its own position and information of the 3D environment. Vision-based Simultaneous Localization and Mapping (SLAM) estimates the location of the robot while incrementally building a map of the environment. However, SLAM only reveals the structural information of the

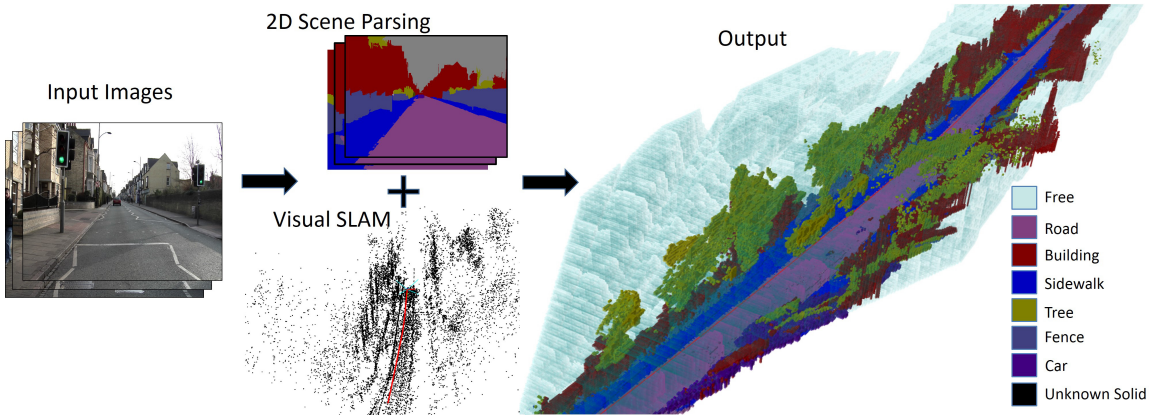

Figure 3.1: Overview of our system. From monocular image sequence, we first obtain 2D semantic segmentation, sparse 3D reconstruction and camera poses. We then build a volumetric 3D map which depicts both 3D structure and semantic labels.

scene and the result is limited to a sparse 3D point cloud. Scene parsing, on the other hand, labels each pixel in an image or video with object categories (*e.g*. *Tree*, *Road*), thus providing semantic only information of the scene. But in many applications such as autonomous driving, it is important to understand both the structural and semantic information of the surroundings. In this paper, we propose a joint 3D reconstruction and scene parsing system from a fast forward-moving monocular camera.

Autonomous driving applications often involve fast forward-moving cameras. In these cases, multi-view stereo could fail due to textureless surfaces and/or low parallax, and the visual SLAM pipeline for a monocular camera only provides a very sparse set of 3D measurements. Previous work on joint reconstruction and scene parsing [63, 64] require dense depth measurements and cannot accommodate to this problem.

Lifting the requirement of dense depth measurements, our input contains only sparse 3D point cloud but dense semantic labels on each pixel of each frame, the latter can be obtained through evaluating a scene parsing engine (e.g. [46]) on all the frames. We use category-specific sensor models to enhance the depth estimates, especially when no direct depth information is available. On the other hand, the knowledge of unoccupied space from successive camera positions help to reduce a lot of 3D structural ambiguities, as well as to improve structural estimates along weakly supported surfaces [65], where only vague structural information is available.

The 3D scene is represented in the form of 3D cubic subvolumes (voxel) along with per-voxel semantic labels (see Fig.3.1). The voxel labels include both solid semantic categories (*e.g*. *Car*) and *Free*, thus capturing both occupancy and semantic information in a single coherent discrete label space. We model the problem of labeling of all observable voxels with a higher order Conditional Random Field (CRF) in the 3D space. Inference of the CRF model in 3D allows for incorporating more realistic scene constraints and priors, such as 3D object support. Besides, full temporal coherency of the semantic labels is inherent to our 3D representation, because our 2D scene parsing is simply the projection of 3D semantic reconstruction to different camera positions. This representation is efficient and compact with an Octree data structure where unused voxels in the 3D map remain uninitialized and require minimal storage and computation.

Our method is applicable to popular monocular sequences like Camvid [66] which has only seen 2D segmentation results till now. Besides, our framework is flexible and can be easily extended to other sensors like laser or stereo cameras. It is quite efficient compared to standard multi-view stereo pipelines and still properly deals with noisy measurements and uncertainty. Thus, our method could find immediate use in many applications like autonomous robot navigation.

3D geometric information plays an important role in 2D semantic segmentation [67, 52, 68, 69]. For example, Brostow *et al*. [67] incorporate sparse SfM features with 2D appearance features for each frame, and demonstrated its advantage over 2D appearance features alone. Ladicky *et al*. [68] propose a joint optimization of dense stereo and semantic segmentation for every frame. However, temporal consistency of the segmentation is not considered in their methods. Several recent attempts [69, 59, 54] have addressed temporal continuity, either by pre-processing with supervoxel-based video segmentation [54], or by additional higher order potentials that enforce label consistency among projections of the same 3D point [69]. Still, most of these methods run in the 2D image space only. Our volumetric representation performs inference in 3D and achieve full temporal coherency

without additional cost.

Semantic segmentation can be used to estimate 3D information [70, 71, 72]. For example, Liu *et al*. [70] guide the 3D reconstruction from a single image using semantic segmentation. Depth from semantics, though not as reliable as the SfM or multi-view stereo, has its own strengths: (1) it is complementary to the traditional geometric approaches; (2) it offers a potential denser depth measurement than SfM; (3) it is applicable for a larger range of sceneries than multi-view stereo. For a fast forward-moving monocular camera, the SfM gives very sparse point cloud and the multi-view stereo fails due to low parallax, whereas we can still rely on segmentation results.

The most relevant work are [63, 64] who have independently proposed methods for simultaneous semantic segmentation and 3D reconstruction. However, both of these methods require dense depth measurements. Dense depth maps allow them to make relatively restrictive assumptions, *e.g*. Haene *et al*. [63] consider every pixel with missing depth as *Sky*. These assumptions do not hold in case of fast forward-moving monocular camera, where we only have a very sparse point cloud from SfM. Unlike [64], we propose a joint optimization scheme of both semantic segmentation and 3D reconstruction. And unlike [63], we use semantic category specific sensor models to estimate the depth as much as possible, instead of simply inserting *Free* labels for voxels with missing depth.

We explicitly model *Free* space. For applications like autonomous driving, *Free* space information is directly used in higher level tasks like path planning. Also, *Free* space provides cues to improve 3D reconstruction, especially along weakly supported surfaces [65] which is very common with forward moving cameras in urban scenes. In our framework, the *Free* space information from other cameras helps to reduce ambiguities in 3D structure.

This paper makes the following contributions:

- From a fast forward-moving monocular camera, we introduce a novel higher order CRF model for joint inference of 3D structure and semantics in a 3D volumetric model. The framework does not require dense depth measurements and efficiently

utilize semantic cues and 3D priors to enhance both depth estimation and scene parsing.

- We present a data-driven category-specific process for dynamically instantiating potentials in the CRF. Our method performs tractable joint inference of 3D structure and semantic segmentation in large outdoor environments.

- We present results on challenging forward-moving monocular sequences such as CamVid and Leuven which demonstrate the value of our approach. The results have shown improved temporal continuity in scene parsing as well as improved 3D structure.

3.2 Problem Formulation and Notation

We are interested in the 3D map $\mathcal{M}$ comprising of several sub-volumes $m_i \in \mathcal{M}$. Where each m_i is a categorical random variable corresponding to voxel i, that can be either *Free* or one of the **solid** semantic objects like *Road*, *Building*, *Tree*, *etc*. For example in the Camvid [66] dataset, we used a 9 dimensional label space $\mathcal{L}_{\mathcal{M}} = \{$*Free*, *Road*, *Building*, *Sidewalk*, *Tree*, *Fence*, *Person*, *Car*, *UnknownSolid*$\}$. Note that this joint label space, $\mathcal{L}_{\mathcal{M}}$ is mutually exhaustive and is different from the label space $\mathcal{L}_{\mathcal{I}}$ of 2D image level semantic categories. For example there is no *Sky* in $\mathcal{L}_{\mathcal{M}}$, a common state used in 2D image scene parsing. Choosing this label space $\mathcal{L}_{\mathcal{M}}$ allows us to do the joint inference of both semantic category and and 3D structure of the scene with a single random variable per voxel.

Each pixel location $x \in \Omega$ in the images is a source of potential measurement, where $\Omega = \{1..h\}\times\{1..w\}$, with $w, h \in \mathbb{Z}^+$ being image size. We have two kinds of measurements : *with-depth* measurements denoted by z^r and *semantic-only* measurements denoted as z^s. Each measurement has an associated semantic label $l \in \mathcal{L}_{\mathcal{I}}$, obtained from the 2D semantic classifier output (§ 3.6.2) at that pixel. Each *with-depth* measurement has an additional depth $d \in \mathbb{R}$ information, which in our case is obtained from visual SLAM (§ 3.6.1).

The observed data is composed of all the measurements and camera poses *i.e.* $\mathcal{D} = \{\mathbf{z}^r_{1:P}, \mathbf{z}^s_{1:Q}, \mathbf{g}_{1:T}\}$, where $\mathbf{z}^r_{1:P}$, $\mathbf{z}^s_{1:Q}$ and $\mathbf{g}_{1:T}$ respectively denotes the set of *with-depth* measurements, *semantic-only* measurements and camera trajectory up-to time T, which in our case is simply equivalent to number of images processed. Each $g_t \in \mathrm{SE}(3)$ is a single camera pose from the camera trajectory. Since we have multiple number of *with-depth* and *semantic-only* measurements per frame, we index them using p and q respectively, where $1 \leq p \leq P$ and $1 \leq q \leq Q$. Also we only have very sparse depth measurements, so $P \mathrm{l} Q$.

We use subscript notation to denote associated camera pose, pixel semantic label, coordinate and depth (if available) for a particular measurement. Thus for a *semantic-only* measurement z^s_q, $l_q \in \mathcal{L}_\mathcal{I}$ denotes 2D image semantic label at pixel coordinate x_q with camera pose g_q. Similarly for p-th *with-depth* measurement z^r_p, d_p encodes the depth of the associated 3D point X_p, measured along the ray emanating from pixel location x_p with semantic label l_p and taken from camera pose g_p. We will sometime drop the superscript in z, when the type of measurement z^r (*with-depth*) or z^s (*semantic-only*) does not matter.

A single measurement z_k only affects a subset of voxels $\mathbf{m}_k \in \mathcal{M}$. For our camera sensor, these voxels are a subset of the voxels lying along the ray emanating from camera center through the corresponding image pixel coordinate of the measurement, denoted as $R_k = \mathrm{Ray}(x_k, g_k)$. Thus the set of voxels affected by a particular measurement z^r_p (or z^s_q) is represented by $\mathbf{m}_p \in R_p$ ($\mathbf{m}_q \in R_q$).

3.3 Probabilistic Model

We utilize a discriminative CRF model on $P(\mathcal{M}|\mathcal{D})$ to avoid directly modeling the complex dependencies [73, 74] among correlated sources of *with-depth* and *semantic-only* measurements. Unlike traditional occupancy grid mapping [75] we do not assume each m_i as independent from each other. Instead, we make use of the standard *static world* conditional independence assumptions of each measurement z_k given the map $\mathcal{M}$, and independence of the map $\mathcal{M}$ w.r.t. the camera trajectory $\mathbf{g}_{1:T}$. Given these assumptions, we can factorize

the posterior over map $\mathcal{M}$ given all the observation data

$$
\begin{aligned}
P(\mathcal{M}|\mathcal{D}) &\propto P(\mathcal{M}|\mathbf{g}_{1:T})P(\mathbf{z}^r_{1:P}, \mathbf{z}^s_{1:Q}|\mathcal{M}, \mathbf{g}_{1:T}) \\
&= P(\mathcal{M}) \prod_{p=1}^{P} P(z^r_p|\mathcal{M},g_p) \prod_{q=1}^{Q} P(z^s_q|\mathcal{M},g_q) \qquad (3.1) \\
&= \underbrace{P(\mathcal{M})}_{\text{prior}} \prod_{p=1}^{P} \underbrace{P(z^r_p|\mathbf{m}_p,g_p)}_{\substack{\text{forward } \textit{with-depth} \\ \text{measurement model}}} \prod_{q=1}^{Q} \underbrace{P(z^s_q|\mathbf{m}_q,g_q)}_{\substack{\text{forward } \textit{semantic-only} \\ \text{measurement model}}} \qquad (3.2)
\end{aligned}
$$

where the conditional independence assumptions were applied to obtain (3.1), and since each measurement is only dependent on a subset of voxels in $\mathcal{M}$, we can further reduce (3.1) to get (3.2). (3.2) uses forward sensor measurement model [75] (measurement likelihood). However, if we adopt this factorization, we would need to learn a complicated sensor model in order to parametrize the forward sensor likelihoods $P(z_k|\mathbf{m_k}, g_k)$. Reapplying Bayes rule on (3.2), we get the inverse sensor model version as

$$
P(\mathcal{M}|\mathcal{D}) \propto \underbrace{P(\mathcal{M})}_{\text{prior}} \prod_{p=1}^{P} \underbrace{\frac{P(\mathbf{m}_p|z^r_p, g_p)}{P(\mathbf{m}_p)}}_{\substack{\text{inverse } \textit{with-depth} \\ \text{measurement model}}} \prod_{q=1}^{Q} \underbrace{\frac{P(\mathbf{m}_q|z^s_q, g_q)}{P(\mathbf{m}_q)}}_{\substack{\text{inverse } \textit{semantic-only} \\ \text{measurement model}}} \qquad (3.3)
$$

which provides the hints that our factors should be similar to posterior probabilities. We can rewrite both (3.2) and (3.3) in terms of factors [76]:

$$
P(\mathcal{M}|\mathcal{D}) = \frac{1}{Z(\mathcal{D})} \underbrace{\psi_\pi(\mathcal{M})}_{\text{prior factor}} \prod_{p=1}^{P} \underbrace{\psi^p_r(\mathbf{m}_p;z^r_p,g_p)}_{\substack{\textit{with-depth} \\ \text{measurement factors}}} \prod_{q=1}^{Q} \underbrace{\psi^q_s(\mathbf{m}_q;z^s_q,g_q)}_{\substack{\textit{semantic-only} \\ \text{measurement factors}}} \qquad (3.4)
$$

where $Z(\mathcal{D})$ is the partition function over the observed data. We now discuss the prior factor and the measurement factors.

3.3.1 Priors

In the above $P(\mathcal{M})$ or the prior factor ψ_π encodes the prior distribution over the huge set of all possible $\mathcal{L}_m^{|\mathcal{M}|}$ maps. However most of these maps are highly implausible and we

can enforce some constraints in form of priors to improve our solution. We enforce the following priors over the map:

- **Spatial smoothness:** Our 3D world is not completely random and exhibits some sort of spatial smoothness.
- **Label compatibility:** Certain pair of classes are more/less likely to occur adjacent to one another. For example a *Car* voxel is unlikely to be adjacent to a *Building* voxel.
- **3D Support:** For most solid semantic categories (with the exception of *Tree*), an occupied voxel increases the chance of the voxels below it to belong to the same occupied category.
- **Free space Support:** *Free* space provides cues to improve 3D reconstruction along weakly supported surfaces [65]. Highly-supported free space boundaries are more likely to occupied.

We model spatial smoothness and label compatibility using pairwise potentials (§ 3.4.4). 3D and Free space support constraints are implemented with unary potentials (§ 3.4.1). Therefore, our $\psi_\pi(\mathcal{M})$ factorizes into pairwise and unary factors.

3.3.2 Measurement Factors

Measurement factors $\psi_r^p(\mathbf{m}_p; z_p^r, g_p)$ and $\psi_s^q(\mathbf{m}_q; z_q^s, g_q)$ encode the constraints imposed by a particular *with-depth* and *semantic-only* measurement respectively. In general, this forms a higher order clique involving multiple voxels $\mathbf{m}_k \subset \mathcal{M}$. However for certain kind of measurements, e.g. *with-depth* measurements or *semantic-only* measurements with *Sky* label, the factor $\psi(\mathbf{m}_k | z_k, g_k)$ can be approximated by a product of unaries on each voxel in $\mathbf{m}_k$. For example when we have a *with-depth* measurement, all voxels along the ray from camera center till the observed depth are more likely to *Free*. And the voxel corresponding to the observed 3D point is likely to belong to a solid semantic category. We use category-specific measurement models (described in § 3.4.2 and § 3.4.3) which can be either unary factors or higher order factors.

3.3.3 CRF Model

As discussed in the above two paragraphs we model the prior factor and the measurement factors in (3.4) with unary, pairwise and higher order potentials. Thus, rearranging the factors in (3.4) in terms of their arity, we get

$$P(\mathcal{M}|\mathcal{D}) = \frac{1}{Z(\mathcal{D})} \prod_{i} \psi_u^i(m_i) \prod_{i,j \in \mathcal{N}} \psi_p(m_i,m_j) \prod_{R \in \mathcal{R}} \psi_h(\mathbf{m}_R) \tag{3.5}$$

Here $\psi_u^i(m_i)$ is the unary potential defined over each m_i, and encodes local evidence. The pairwise potential, $\psi_p(m_i,m_j)$ over two neighboring voxels falling into a neighborhood $\mathcal{N}$ enforces spatial smoothness and label compatibility among them. Higher order cliques $\psi_h(\mathbf{m}_R)$ are defined over set of voxels $\mathbf{m}_R$ along some ray emanating from a 2D image projection and helps with missing depth information. Fig.3.2(a) shows the corresponding factor graph $\mathcal{H}$ of the model.

A single *semantic-only* measurement z_q^s for certain classes is ill-posed for updating states of the affected voxels $\mathbf{m}_q$ since we do not know which voxel reflects back the measurement. Häne *et al*. [63] simply updates all $\mathbf{m}_q$ with *Free* unaries for measurements missing depth, which is clearly an improper model. In our approach, we handle such measurements without range/depth, by forming higher order factor connecting voxels along a ray. However a naive approach will lead to forming huge higher order cliques and since every pixel in every image is an potential measurement, and inference in the graphical model can become intractable very soon. To circumvent this issue, whenever applicable, we make use of semantic cues to model them with unaries or at least reduce the scope of such higher order factors.

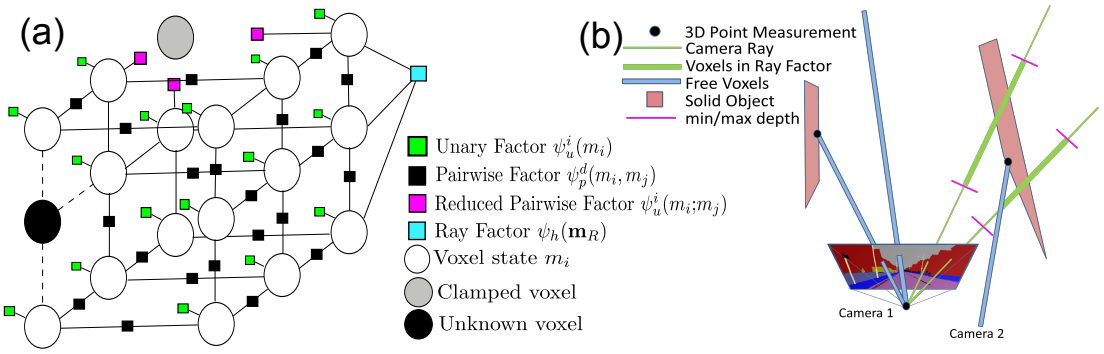

Figure 3.2: (a) Factor Graph $\mathcal{H}$ of our framework. (b) Illustration of sensor models and higher order Ray factors. See text for more details.

3.4 Potentials

3.4.1 Basic Unary Potentials

We have different types of measurements, and they affect m_i differently. For example 3D depth measurement alone do not contain any semantic label information and influence all semantic label probabilities equally. Also each category of semantic observation affects the belief state of a voxel m_i, differently than others. We define the following two basic forms of unary measurement factors:

$$\psi_{\text{MISS}}(m_i) = \begin{cases} 0.6 & \text{if } m_i = \textit{Free} \\ \frac{0.4}{|\mathcal{L}_\mathcal{M}|-1} & \text{if } m_i \neq \textit{Free} \end{cases} \quad \text{and} \quad \psi^l_{\text{HIT}}(m_i) = \begin{cases} 0.3 & \text{if } m_i = \textit{Free} \\ 0.55 & \text{if } m_i \equiv l \\ \frac{0.15}{|\mathcal{L}_\mathcal{M}|-2} & \text{if } m_i \notin \{l, \textit{Free}\} \end{cases} \tag{3.6}$$

Fig.3.3 illustrates the measurement factors ψ_{MISS} and $\psi^{\textit{Road}}_{\text{HIT}}$. Note that, we have made use use of inverse sensor model $P(m|z, g)$ for these factors. This is motivated by the fact that, it is much more easier [75] to elicit model parameters for $P(m|z, g)$ compared to the forward sensor likelihoods $P(z|m, g)$, and can be done without resorting to complicated sensor model learning. We kept the parameters same as that of laser based occupancy sensor model used in [77].

The unary potential $\psi^i_u(m_i)$ combines all the unary measurement factors that affect m_i. Thus the final unary potential over a voxel is factor product of a certain number of ψ_{MISS}

and ψ^l_{HIT} factors only.

$$\psi^i_u(m_i) = [\psi_{\text{MISS}}(m_i)]^{N_M} \prod_{l \in \mathcal{L}_\mathcal{I} \setminus Sky} [\psi^l_{\text{HIT}}(m_i)]^{N_{Hl}} \tag{3.7}$$

where N_M is the total number of MISS unary factors over m_i and N_{Hl} being the number of HIT factors over m_i for semantic category l. Fig.3.3(c) depicts the factor graph view of this potential.

As new measurements are obtained, we keep on inserting new factors into the affected voxels. The set of voxels affected, and the kind of unary factors that gets inserted depends on the measurement type (discussed in next two subsections).

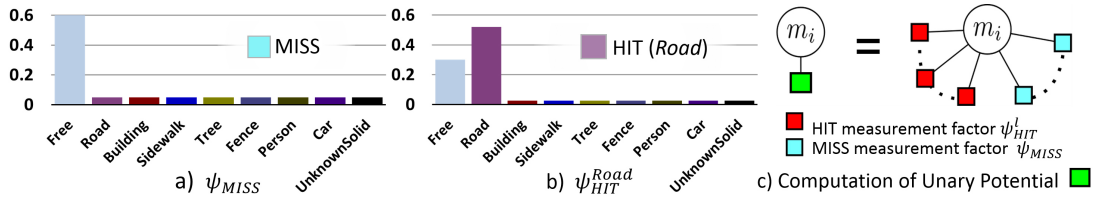

Figure 3.3: a) and b) illustrates the MISS and HIT factors. c) Computation of per voxel unary potential as a product of unary contributions of several measurements affecting that voxel.

3.4.2 Measurements with depth

We use a projective camera sensor model, wherein the basic assumption is that each measurement is formed by reflection from a occupied voxel at some particular depth, and all voxels from the camera center to that depth are *Free*. So for all voxels from camera center till the observed depth, we insert a MISS factor which increases the probability for these voxels being *Free*. And for the voxel corresponding to the observed 3D point X_p, we insert a HIT factor which makes the probability of belonging to a particular solid semantic state high. Our framework is not limited to monocular only system, the same approach can also be extended to a Laser+Vision system, where measurements from lasers affect all solid semantic category probabilities equally.

3.4.3 Semantic only Measurement

With sparse reconstruction most points in the image do not have direct depth measurements. However certain classes of measurements still provide a good estimation of depth. Observing *Sky* tells us that all voxels along the observed ray are more likely to be *Free*. Fig.3.4 LEFT shows average depth for some semantic categories across different parts of the image. We computed these statistics on the sequence seq05VD of Camvid. We first form a uniform 2D grid over the image, and then for each such grid in the image, we accumulate the depths from visual SLAM point clouds whose projection on the image lie on that grid. This gives us information about how good a *semantic-only* measurement z_q^s is in estimating the 3D depth. For each semantic class, all measurements with 2D projection x lying on the same grid gets same statistics. Two kind of statistics are computed for each such possible $(l_q, x_q) \in \mathcal{L}_\mathcal{I} \times \Omega$ measurement. The *min* depth and *max* depth for each (l_q, x_q) tells us the minimum and maximum possible depth along pixel co-ordinate x_q for 2D semantic category l_q. We then also estimate inverse sensor model $P(\mathbf{m}_p|z_q^s, g_q)$. Fig.3.4 shows the plots of inverse sensor model along with min/max depth for two specific *semantic-only* measurements, $(l_q = \textit{Road}, x_q = [400, 700])$ and $(l_q = \textit{Building}, x_q = [100, 300])$. When the statistics shows a small min-to-max bound *e.g. Road* and the inverse sensor model has a high peak, we insert unary factors according to this inverse sensor model.

However for certain classes like *Building*, depth uncertainty is too high to make it effective, since they can occur at different depths. Using unaries for these measurements introduces a lot of artifacts. So for these class of *semantic-only* measurements we construct a higher order factor involving all the voxels along the ray that lie between *min* depth and *max* depth computed for that semantic measurement. Solid *semantic-only* measurements like *Building*, *tree*, even though does not say much about the depth, confirms the fact that there is at least one occupied voxel along the ray induced by that observation. Our **Higher**

order Ray Potential simply encodes this fact and can attain only two possible values:

$$\psi_h(\mathbf{m}_R) = \begin{cases} \alpha & \text{if atleast one of } \mathbf{m}_R \text{ is } \neg \textit{Free} \\ \beta & \text{if all of } \mathbf{m}_R \text{ is } \textit{Free} \end{cases} \tag{3.8}$$

where $\mathbf{m}_R$ is set of voxels along a particular ray involved in the factor and $\alpha > \beta$. We make use of the class specific prior knowledge of the minimum depth and maximum depth of the reflecting voxel along a particular 2D back-projection. So for a ray factor $\psi_h(\mathbf{m}_R)$ caused by a measurement z_q^s, $\mathbf{m}_R = \{m_i : m_i \in R_q, min(l_q, x_q) \leq \text{depth}(m_i, g_q) \leq max(l_q, x_q)\}$. This reduces the number of voxels $|\mathbf{m}_R|$ involved in $\psi_h(\mathbf{m}_R)$, which could otherwise be very large (see Fig.3.2(b) for illustration). A further reduction is facilitated by strong free space measurements (see § 3.5.3). In contrast, the higher order factors used in [78] involve all the voxels starting from the camera. Another contrast to [78] is that our ray factor captures single view constraints which is orthogonal to multiview higher-order factors of [78] requiring costly photoconsistency computations across multiple views. Note that the higher order factor (3.8) is a sparse one and its of the same form as $\mathcal{P}^n$ Potts model [79] (a special case of Pattern potentials[80]) which allows us to do tractable inference (§ 3.6.3).

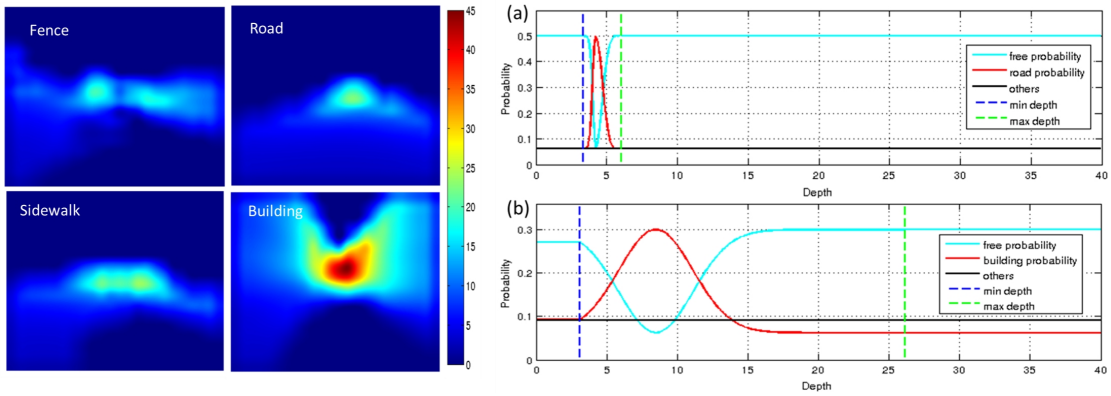

Figure 3.4: LEFT: Average per category depthmap of Camvid [66] (subsequence # seq05VD) for *Fence*, *Road*, *Sidewalk* and *Building*. RIGHT: shows the inverse sensor model $P(m_i|z_q^s, g_q)$ for voxels i along the ray emanating from 2D point x_q as function of depth from camera center. (a) shows the inverse sensor model for a *Road* point measurement at 2D point co-ordinate, $x_q = [400, 700]$. (b) row shows the inverse sensor model for a *Building* observation at point $[100, 300]$. The plots also shows the min and max depth for these measurements.

3.4.4 Spatial smoothness and Label compatibility

The pairwise factor $\psi_p^d(m_i, m_j)$ enforces spatial smoothness and label compatibility between pairs of neighboring voxels defined by 3D neighborhood $\mathcal{N}$. Thus each voxel can have a maximum of 26 pairwise factors. The pairwise factors ψ_p^d are also dependent on relative direction d (horizontal or vertical) between the voxels. This allows us to capture properties like *Road* or *Sidewalk* voxels are more likely to be adjacent to each other in horizontal direction. So our pairwise potential is like Potts model, except that we set different weights for certain specific pairs of labels. To prevent *Free* voxels encroach other solid voxels, we set a lower cost for a $\psi_p^d(m_i = Free, m_j \neq Free)$ than other pairs in $\mathcal{L}_\mathcal{M} \times \mathcal{L}_\mathcal{M}$.

3.5 Data-driven Graphical Model Construction

The final graphical model is dynamically constructed and fully specified once all unary potentials has been computed.

3.5.1 Data Structure for Scene Representation

We use an octree based volumetric data structure which provides a compact storage of the scene. In the octree representation, when a certain subvolume observes some measurement, the corresponding node in the octree is initialized. Any *uninitialized* node in the octree represents *Unknown* areas. *Unknown* voxels are not included in the space over which we construct the graphical model and run our inference algorithm. This is different than other common approaches [78, 63] of inferring over all voxels within a bounding box.

Of all factors used in our model, only the unary factor ψ_u^i is of different values for every m_i. All other factors like pairwise factors ψ_p or higher order ray factors ψ_h even though has different scopes, are fixed functions and we need to just store only **one instance** of them. Each node of the octree stores the local belief $bel(m_i)$ (as *log* probabilities) which

is equal to the prior probability at time zero, and is incrementally updated to yield the final unary factor $\psi_u^i(m_i)$. Thus unlike a naive approach, we do not need to explicitly store all measurements, which is huge even for a short video sequence. Also note that all other factors apart from ψ_u^i are either precomputed, can be computed directly from voxel co-ordinates or from ψ_u^i itself without needing access to the raw measurement data.

3.5.2 Clamping

Even for nodes which have been initialized, if the local belief $bel(m_i)$ for a particular state $\in \mathcal{L}_\mathcal{M}$ has reached a very high probability (we used 0.98), we fix m_i to that state and treat it like evidence. This clamping of voxels which are already very confident about its label, reduces the total number of variables involved in the inference and also the scope of pairwise/higher-order factors attached to them. A pairwise factor between m_i and m_j gets *reduced* to unary factor $\psi_u^i(m_i) = \psi_p(m_i, m_j = \textit{Free})$, when m_j gets clamped to *Free* label. In Fig.3.2(a), the shaded node ● represents such a clamped voxel and ■ denotes the reduced pairwise factors. Clamping of confident voxels and conservative generation of set of voxels over which we do the final inference, allows us to scale to longer sequences and not just scenes with a small fixed bounding box.

3.5.3 Scope Reduction of Higher Order Ray Potentials

Since the final graphical model structure $\mathcal{H}$ is computed only after all the unary potentials have been computed, it allows for further reduction of number of voxels $|\mathbf{m}_R|$ involved in higher order ray factors (3.8). We illustrate this with help of Fig.3.2(b). Suppose Camera1 receives a *semantic-only* measurement, which results in a higher order ray factor involving voxels lying between min and max depth for that measurement. But strong free space measurements coming from other cameras (*e.g.* Camera2 in Fig.3.2(b)) helps us in further reducing the number of voxels $|\mathbf{m}_R|$ in the scope of that ray factor.

3.5.4 3D support and Free space support

Most solid semantic categories (with exceptions *e.g*. *Tree*) have a 3D support, as in an occupied voxel increases the chance of the voxels below it to belong to the same occupied category. So for voxels which have been clamped to semantic categories like *Building*, *Fence*, *Pole*, we insert a extra HIT unary factor corresponding to the same semantic category for all voxels lying directly below.

As shown by [65], highly-supported free space boundaries are more likely to be occupied. This is important for driving sequences, since most surfaces like road are very weakly supported by measurements. For voxels for which have been clamped to *Free*, we first check if there are *Unknown* voxels directly adjacent to it. If upon back-projecting these *Unknown* voxel coordinates to the images, we get a strong consensus in a solid semantic label: we initialize that voxel node and insert a single HIT unary factor corresponding to that label.

3.6 System Pipeline

With input monocular images, we first perform visual SLAM and an intial 2D scene parsing using standard semantic segmentation methods [46, 32]. We then do a data-driven graphical model construction (§ 3.5.1) based on these measurements, followed by a final inference step.

3.6.1 Visual SLAM

Visual SLAM estimates the camera trajectory $\mathbf{g}_{1:t}$ and sparse 3D point cloud $\{X\}$ where $g_t \in \text{SE}(3)$ and $X \in \mathbb{R}^3$. We do frame-to-frame matching of sparse 2D feature points, followed by RANSAC based relative pose estimation to obtain an initial estimate of the camera poses. A further improvement in feature tracking is obtained by rejecting matches across a image pair if the matched points lie on areas labeled as different semantic cate-

gories by the 2D semantic classifier. Finally we use bundle adjustment [81, 82], which iteratively refines the camera poses and the sparse point cloud by minimizing a sum of all re-projection errors. Once bundle adjustment has converged, we obtain a set of sparse 3D points and corresponding camera poses from which each of these points have been observed.

3.6.2 Initial 2D Scene Parsing

We use the unary potentials used by Ladicky *et al*. [46] consisting of color, histogram of oriented gradients (HOG), pixel location features and several filter banks. We then use the dense CRF implementation of [32] to get the baseline 2D scene parsing. Since we directly work from per pixel semantic labels, any other scene parsing method can be used instead.

3.6.3 Inference Algorithm

For doing inference over the graphical model, we use the maximum a-posteriori (MAP) estimate $\mathcal{M}^* = \arg\max_{\mathcal{M}} P(\mathcal{M}|\mathcal{D})$ to assign a label to to each m_i. The rationale behind MAP is the big progress [83] of efficient approximate MAP inference in recent years. We use a modified message passing implementation of [83]. We use tree-reweighted (TRW) [84] messaging schedules. For computing messages to and from the higher order factors (3.8) we use the approach of [85]. Since our higher order factors (3.8) are sparse, all n outgoing messages from these higher order factors can be computed in $\mathrm{O}\big(n\big)$ ($\mathrm{O}\big(1\big)$ amortized) time.

3.7 Experiments and Evaluation

Since we are jointly estimating both 3D structure and semantic segmentation, it is expected that we improve upon both of them. In this section we define the evaluation criteria for measuring the above and show results to verify our claim. We demonstrate results of our method on Camvid [66] and Leuven [86, 68] datasets. Both these datasets involve difficult

fast forward moving cameras and has been standard dataset for semantic segmentation papers [67, 68, 54, 59, 69]. Leuven dataset contains stereo image pairs, but we demonstrate results only using monocular (left) images. To the best of our knowledge, we are not aware of any other work which has demonstrated joint 3D reconstruction and semantic segmentation on these standard monocular datasets. We additionally provide results on small sub-sequence of KITTI [25], again using monocular (left) images.

3.7.1 3D Structure Quality

We vastly improve upon the baseline 3D structure estimated through traditional SfM approach. Fig.3.6 shows some of our 3D reconstructions of a part of Camvid [66]. Note the improvement obtained over state of the art multi-view stereo [87] and sparse SfM in Fig.3.6. In the Leuven sequence, shown in Fig.3.5, we compare against the stereo based 2.5D method of Ladicky *et al*. [68] for joint segmentation and stereo. We back-project our 3D semantic map onto the cameras to obtain per frame depth/disparity image. Fig.3.5 qualitatively demonstrates the better quality of our 3D structure estimate, both in comparison to the stereo disparity maps and to baseline sparse SfM, even though only monocular(left) images were used compared to stereo method of [68]. In Fig.3.7, we compare against unary-only results with LIDAR sensor in KITTI [25].

3.7.2 Segmentation Quality

From our 3D joint semantic map, we can obtain 2D segmentation result by simply back-projecting it to each camera views. We evaluate segmentation quality in terms of both per pixel segmentation label accuracy and also temporal consistency of the segmentation in videos. We achieve significant improvement in both the measures over state of the art. To evaluate temporal consistency, we first select a set of confident SfM feature tracks which has very low re-projection errors after bundle adjustment. So these static 3D points should ideally be having same label from all the images it is visible from. So lower entropy (less

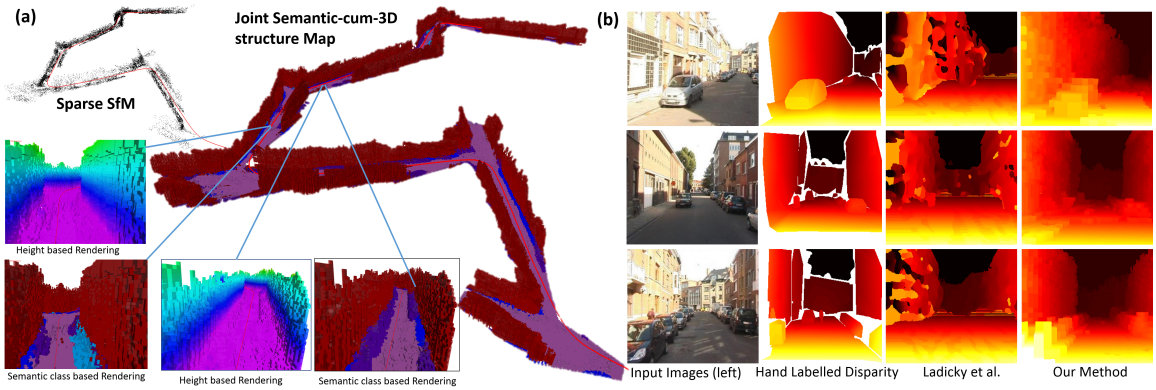

Figure 3.5: Leuven [68] Results. (a): the output semantic reconstruction of the Leuven sequence, using only left (monocular) images. *Free* voxels are not shown for clarity. Note the improvement compared to initial SfM pointcloud. (b) Comparisons with the stereo method of Ladicky *et al*. [68], by using monocular (*left*) images *only*. We obtain 2D depth maps by back-projecting our 3D map onto the cameras. Notice the significant improvement over the depth maps of [68] when compared to the hand labeled disparity image provided by [68].

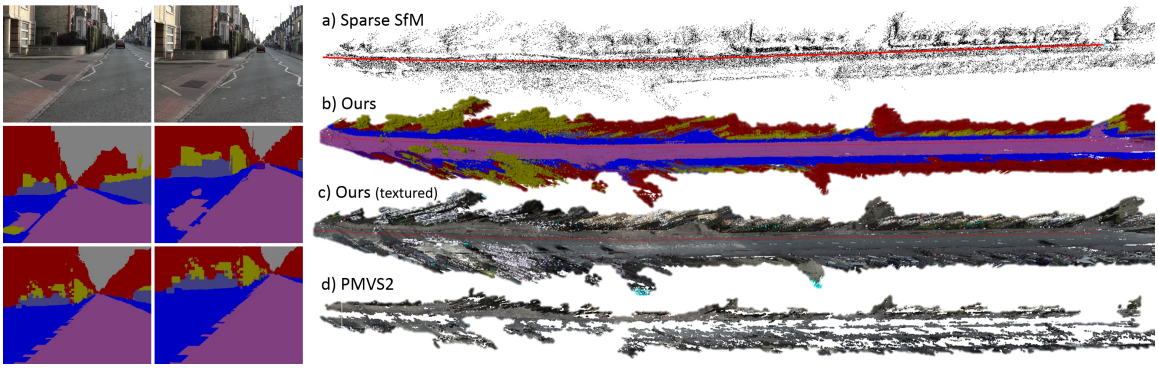

Figure 3.6: CamVid [66] Results. LEFT: Top row shows two consecutive input images, middle row shows baseline 2D segmentation and bottom row shows 2D segmentation obtained by back-projecting our 3D semantic map. Note the temporal inconsistency in baseline 2D segmentation (middle row). RIGHT: a) 3D reconstruction and camera trajectory from Visual SLAM. b) Our 3D semantic + occupancy map using the same legend as in Fig.3.1. *Free* voxels are not shown for clarity. c) shows the same map, but textured. d)Reconstruction result by PMVS2 [87]. Note the improvement in our map (b,c) compared to sparse SfM(a)and PMVS2(d).

CAMVID seq05VD	*Building*		*Road*		*Car*		*Sidewalk*		*Sky*		*Tree*		*Fence*		*All*	
	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)
Ours	**0.0**	98.3	**0.0**	97.8	**0.0**	95.8	**0.0**	**98.3**	NA	99.3	**0.0**	**83.6**	**0.0**	73.7	**0.0**	**95.5**
[46]	0.114	**98.5**	0.024	96.0	0.231	89.4	0.177	96.5	NA	**99.8**	0.168	83.0	0.299	**75.6**	0.095	94.6
[59]	0.114	94.8	0.016	98.8	0.106	99.7	0.184	94.1	NA	99.2	0.173	80.3	0.249	39.1	0.084	92.4
[54]	0.025	95.0	0.004	**99.0**	0.046	**99.9**	0.062	73.2	NA	99.3	0.037	74.1	0.107	4.4	0.019	87.9
LEUVEN	*Building*		*Road*		*Car*		*Sidewalk*		*Sky*		*Bike*		*Pedestrian*		*All*	
	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)
Ours	**0.0**	**96.5**	**0.0**	**99.4**	**0.0**	**91.8**	**0.0**	67.0	NA	**95.3**	**0.0**	83.8	**0.0**	NA	**0.0**	**95.7**
[68]	0.046	95.8	0.116	98.8	0.150	91.4	0.429	**74.9**	NA	93.3	0.264	**84.7**	0.686	**61.8**	0.094	95.2
KITTI seq05	*Building*		*Road*		*Car*		*Sidewalk*		*Sky*		*Tree*		*Fence*		*All*	
	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)	H(bits)	Acc(%)
Ours	**0.0**	**98.9**	**0.0**	**98.7**	**0.0**	97.0	**0.0**	**98.4**	NA	99.4	**0.0**	96.4	**0.0**	**96.3**	**0.0**	**97.2**
[46]	0.165	97.5	0.113	87.8	0.203	**98.1**	0.158	96.0	NA	**99.8**	0.129	**97.5**	0.220	91.6	0.163	95.2

Table 3.1: 2D Segmentation evaluation. For evaluating temporal consistency, we give average Entropy H of SfM feature tracks (See § 3.7.2). Our results gives perfect zero entropy compared to non-zero entropy (indicating temporal inconsistency) for [59, 54, 68, 46]. We also show the per pixel label accuracy. We again obtain the best results. Best scores has been **highlighted**.

changes in labels) for these SfM feature tracks is an indication of better temporal consistency. Table 3.1 shows the entropy scores for several state of art methods[59, 68, 54, 46] where a higher entropy (in *bits*) indicates more temporal inconsistency. As a consequence of our model and 3D representation we achieve *perfect* consistency. We also evaluate per-pixel label accuracy and as shown in Table 3.1, our method achieves a noticeable gain over state of the art. The supplementary material has more discussion on these results.

3.8 Conclusion

We presented a method for joint inference of both semantic segmentation and 3D reconstruction, and thus provides a more holistic 3D understanding of the scene. Our framework offers several advantages : (a) Joint optimization of semantic segmentation and 3D reconstruction allows us to exploit more constraints and apply more informed regularization achieving improvement in both the tasks; (b) The 3D graphical model allows to incorporate more powerful 3D geometric cues compared to standard 2D image based spatial smoothness constraints; (c) It works for difficult forward moving monocular cameras, where sparse SfM is the only robust reconstruction method, and obtaining dense depth maps (required

by [63, 64]) is difficult; (d) We obtain full temporally consistent segmentations, without ad hoc constraints as in other 2D video segmentation methods [69, 59, 54]; (e) The output is in the form of a 3D volumetric semantic + occupancy map, which is much more useful than a series of 2D semantic label images or sparse pointcloud and it thus finds several applications like autonomous car navigation.

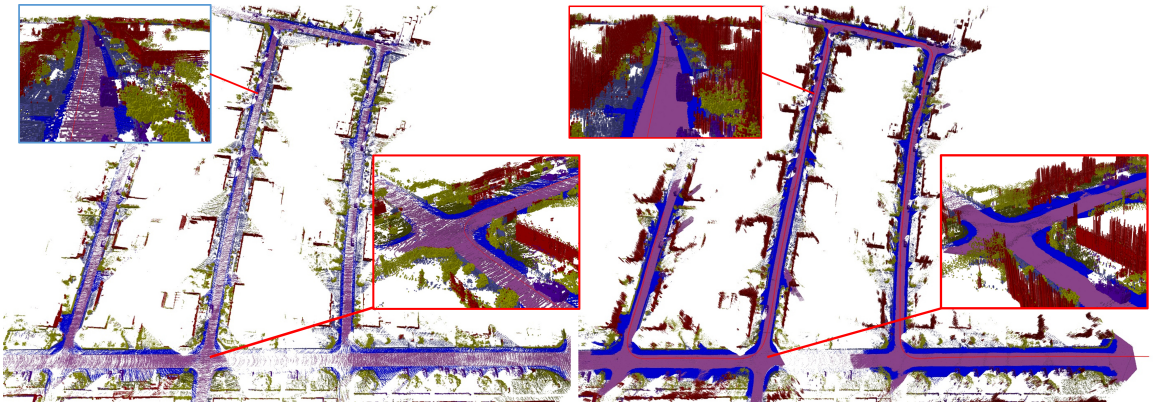

Figure 3.7: KITTI [25] Results (seq 05). LEFT: We use LIDAR measurements available in KITTI using only the unary potentials described in this paper. RIGHT: Results with monocular (left) images and our full CRF model. As can be seen in the figure, even with just monocular images, we are able to achieve more complete reconstruction. For fair comparison, we only used those laser rays from the 360° LIDAR that can be seen by the left camera.

4

INSTANCE LEVEL 3D SCENE UNDERSTANDING

4.1 Introduction

The "scene understanding" term has been used in computer vision to broadly describe high-level understanding of the image content. A scene understanding algorithm builds a compact representation of the image that is easily accessible to subsequent tasks. Traditional scene understanding algorithms has been mostly used to assign semantic labels to pixels in an image or to output 2D boxes around objects of interest. However such 2D representations are insufficient for tasks like planning and 3D spatial reasoning. In this chapter, we argue for a rich 3D scene model which reasons in terms of object instances.

A 2D image is a formed by a complex function of several attributes like lighting, shape and surface properties of objects in the scene. Unlike 2D projection, an instance level 3D understanding provides a disentangled representation of the scene. This disentangled 3D representation makes our method more suitable for real-world applications. The output from our system can be directly used for tasks like path-planning, accurately predicting a object's 3D location in future under some physics model. Another major benefit of doing scene understanding with a rich 3D scene model is that, traditional 2D scene representations like segmentation, detection box, and 2D depth-maps are all available for free. They can be generated by simply rendering the output 3D scene model. But how do we invert the complex image formation process to obtain the 3D scene model?

One classical approach to solving inverse problems in vision is *analysis-by-synthesis*. It consists of using a model that describes the observed data generation process (synthesis), which is then used to estimate the parameters of the model that generated the particular

observed data (analysis). *Analysis-by-synthesis* with a 3D scene model is like **"solving vision as *inverse-graphics*"**. *Synthesis* describes the process of generating image content from the 3D scene model in the style of *computer graphics*. Vision is then like doing *analysis* by searching the best 3D scene configuration to explain the observed image.

The idea of *analysis-by-synthesis* can be rooted back to Helmhotz's 1867 work on unconscious inference [5]. The *analysis-by-synthesis* approach has a long history [8, 9, 10, 6, 7]. While conceptually elegant, it has only been successful for very limited problems. This is due to the fact that typical 3D scene representation is very high-dimensional. So analysis then becomes a difficult search problem over this large, high-dimensional scene variables. Moreover, the posterior distribution of the scene variables is usually very multi-modal and could easily lead an optimization algorithm to get stuck in local minima.

Recently there has been a re-emergence of the *inverse-graphics* approach [16, 17, 18, 13, 19, 20, 15, 21], in which an efficient, discriminative bottom-up method like a convolutional network is used to cut down on the search space. However most of these approaches are still restricted to simple scenes often containing only one object. In this work we present an *inverse-graphics* approach which is capable of handling complex real-world 3D scenes. Our approach uses a deep convolutional network to map image regions to 3D representations of all object instances in an image.

To make the inverse graphics approach scale to complex scenes, we make some key design choices discussed below:

(i) Rather than having separately trained models for bottom-up and inference stages [15, 13, 21, 19], we employ a single unified end-to-end trained network for *inverse-graphics*. We propose a differentiable Render-and-Compare loss, that allows the bottom up process to also obtain supervision from 2D annotations.

(ii) We factorize the scene into object instances with associated shape and pose, so the network can be bootstraped with direct 3D supervision of shape and pose whenever such data is available. This helps with network convergence. Our method provides

a disentangled representation (shape and pose) of an object instance by design. In contrast, other methods [17] have to explicitly train the network to encourage disentanglement and interpretability in latent parameters. We do not explicitly model lighting and material properties, which are nuisance parameters for our intended application of autonomous driving.

(iii) We exploit rich shape priors by learning a class-specific low-dimensional embedding of shapes from CAD model collections [88, 89]. The low-dimensionality of shape-space makes the learning task easier and allows for efficient back-propagation through Render-and-Compare loss. Additionally the shape prior enables a complete (amodal) reconstruction of an object, even for parts of the object which are occluded.

(iv) We carefully study equivariance [90, 91, 92] demands for predicting 3D shape and pose from an image region of interest (RoI). Since shape and pose are 3D entities, normalization of these parameters w.r.t. to 2D RoI transformations is not possible in the same manner it is done for 2D entities like bounding box parameters and instance segmentation [90]. Instead we capture the 2D transformations performed by RoI pooling layers and feed them to shape and pose classifiers.

Our contribution is a fast *inverse-graphics* network called 3D-RCNN, capable of estimating the amodal 3D shape and pose of all object instances in an image. Our method achieves state-of-art performance in complex real world datasets of PASCAL3D+ [26] and KITTI [25].

4.2 Related Work

There have been several works for instance-level 3D scene understanding [93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107]. However most of these approaches [100, 99, 98, 97] only predicts the object orientation. When it comes to shape, most methods either estimate only 3D bounding boxes [93, 95, 107], or coarse wire-frame

skeleton [96, 108, 105, 106], or represents shape via an exemplar mesh chosen from a small set of meshes [94, 101, 103, 104]. In contrast, we jointly learn the detailed 3D shape along with pose. We make use of compact parametric shape-space which has much more capacity than a small set of exemplar meshes and can even represent articulated objects.

There are also several works devoted specifically to shape modeling: learning shape via auto-encoders [109, 110, 111], generative adversarial networks [112], and non-linear dimensionality reduction [113, 114]. In this paper, we chose to adopt PCA for modeling rigid objects, since it is simple and efficient. Our method is flexible to other parametric shape model including articulated shapes, provided it is continuous and relatively low dimensional. We demonstrate the use of SMPL [115] shape model for articulated persons.

Modern rasterization rendering like OpenGL are fast, but lacks a closed-form expression which makes it harder to compute derivatives. It is also discontinuous at occlusion boundaries. However the recent works [116, 15, 117], have demonstrated efficient ways of obtaining approximate derivatives. Chain-rule along with screen-space approximation around occlusion boundaries is used in [116], while [15] uses numerical derivatives. However both these approaches [116, 15] used differentiable rendering in context of test-time optimization for refining certain task parameters, initialized from a separately trained learning algorithm. We also use numerical derivatives, but we use it for computing gradients to back-propagate an end-to-end learned deep convolutional network. In the unsupervised shape reconstruction work of Rezende *et al*. [117], gradients of an OpenGL renderer were computed using [118]. However it was only demonstrated for very simple meshes.

A good majority of the related approaches like [96, 93, 119, 108, 99, 100], process only a single object at a time. This requires multiple passes of their network to cover all objects in the image, which is prohibitively expensive. Our method computes the 3D shape and pose of all objects within a single forward pass of the network, and does not involve any costly post-processing step. With a ResNet-50 backend, our model reconstructs the 3D shape and pose of all object instances in an image in under 200ms and is thus suitable for

realtime applications like autonomous driving.

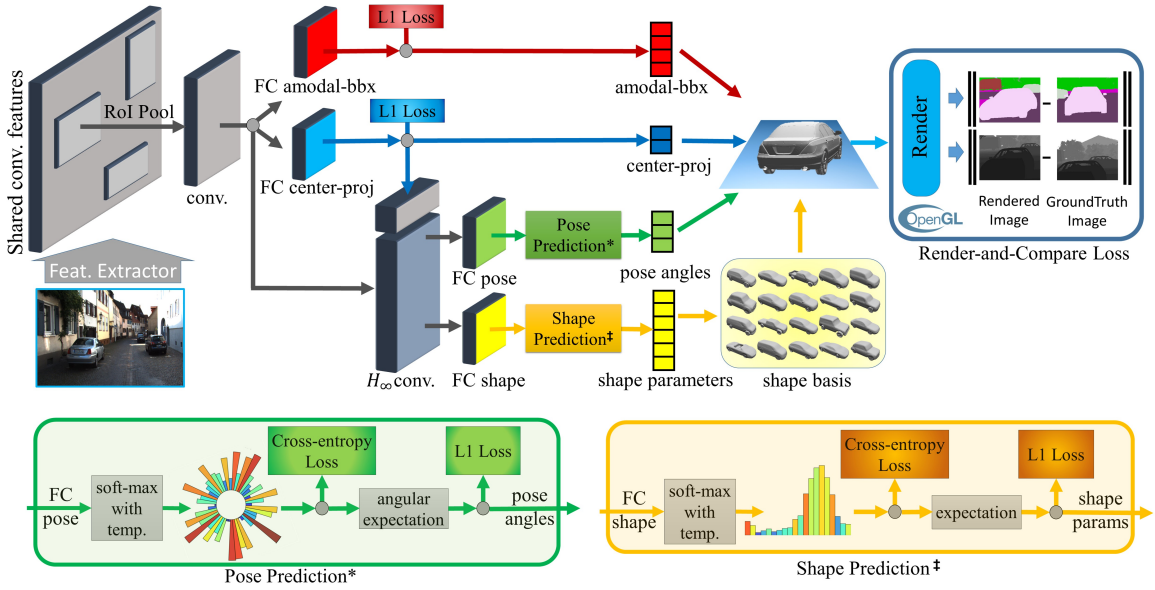

Figure 4.1: Our network architecture for instance-level 3D scene understanding from images. We use ResNet-50-C4 [120] as backbone feature extractor. Layers colored in **gray** are shared across classes. Render-and-Compare loss is described in § 4.5.3. H_∞ convolution is described in § 4.5.1. Shape and Pose prediction modules are expanded in the right and described in § 4.5.2.

4.3 Method Overview

Our goal is to recover the 3D shapes and poses of all object instances within an given image. We assume that object category detector outputs are given, and focus on the extremely challenging task of recovering the 3D parameters of object instances from their 2D observations. A basic challenge which must be addressed is how to represent shape and pose in 3D. We encode object shape using a class-specific shape prior– a low-dimensional "shape" space constructed from a collection of 3D CAD models. This representation encodes common 3D shapes in the object class using a small set parameters. The problem of estimating shape is then casted to the problem of predicting an appropriate set of low dimensional shape parameters for a particular object instance.

We use a learning-based approach to solve the inverse problem of recovering a 3D object representation from 2D pixel observations. Specifically, we learn deep networks that

map image regions to the 3D object parameters defined by our shape and pose representations. Since the final pose and shape prediction are done on the fixed-size feature-map cropped from a region of interest (RoI), it is important to re-parametrize traditional egocentric object pose representation, to an allocentric one. Equally important is to not ask the network to directly predict the location (distance) of the object, since it is a fundamentally ill-posed problem. We present our novel object pose representation in § 4.4.2. We make two key technical contributions. First, we leverage a *differentiable* render-and-compare operation so that we can exploit large-scale existing datasets of image-based object properties such as segmentation masks during training. Second, we achieve equivariance in shape and pose estimation by modeling the geometric distortion induced by pooling and providing it to the network using dynamic filters. The resulting network for 3D shape and pose estimation from 2D image regions is trained end-to-end, and can learn from both synthetic and real image data. Once trained, the model requires only a single very efficient forward pass to obtain accurate shape and pose. Fig. 4.1 presents on overview of our method. We now detail our 3D representation and our network design.

4.4 3D Object Instance Representation

An object instance is fully described by the object's pose and shape.

4.4.1 Shape Representation

3D models in standard mesh or volumetric representations are very high dimensional. However, object instances belonging to the same category tend to have similar shapes. We make use of rich shape priors available in the from of large collections of 3D CAD models [89, 88]. We assume that 3D shapes of instances from a same object category lies in a much lower dimensional manifold. We exploit this by learning a class-specific, low dimensional shape embedding space from collection of 3D CAD models. With the learned embedding, the problem of reconstructing shapes is simplified to finding the corresponding point in the

low dimensional embedding space that best describes observed data.

Given a collection of CAD models [89, 88], we first axis-align them to a common rest pose. We also normalize the shape vertices, such that longest diagonal is of unit length. Since CAD models in mesh representations have arbitrary dimensionality and topology, we convert each model to a volumetric representation $\boldsymbol{s} \in \mathbb{R}^n$ with fixed number of voxels n. Each voxel in the volumetric representation $\boldsymbol{s}$, stores truncated signed distance function (TSDF) [121]. So the surface is implicitly defined by the zero level-set of the TSDF volume.

Given a collection of t TSDF volumes, $\mathcal{S} = [\boldsymbol{s}_1, \ldots, \boldsymbol{s}_t]$ generated from CAD mesh models, we use PCA to find a ten dimensional shape basis, $\mathcal{S}_\mathcal{B} \in \mathbb{R}^{n \times 10}$. Since n is very large and $n >> t$, it is important to use dual form of PCA [122]. Once we have learned $\mathcal{S}_\mathcal{B}$, any TSDF shape $\boldsymbol{s}$, can be encoded to the low dimensional shape parameter $\boldsymbol{\beta} = \mathcal{S}_\mathcal{B}^T \boldsymbol{s}$. Likewise, given shape parameters $\boldsymbol{\beta} \in \mathbb{R}^{10}$, we can decode it to get back to TSDF space as $\boldsymbol{s} = \mathcal{S}_\mathcal{B} \boldsymbol{\beta}$. Some points from our learned shape space of cars and motorcycles are shown in Fig.4.2. We train our network to predict this low dimensional shape parameter $\boldsymbol{\beta} \in \mathbb{R}^{10}$ from images.

There are several different methods for modeling 3D shape space [113, 114]. We chose to adopt PCA since it is simple and efficient. Our method is flexible to any other parametric shape model including articulated shapes, provided it is relatively low dimensional. We demonstrate the use of SMPL [115] for articulated persons in addition to parametric TSDF shape-space described above for rigid objects.

4.4.2 Pose Representation

We are interested in obtaining pose parameters of each object instance in the full-image camera frame. This includes object root pose $\mathcal{P}_\mathcal{E} \in$ **SE(3)**, made of an object's 3D orientation and position. For articulated objects, this includes additional joint angles $\boldsymbol{j}$ relative to the root pose $\mathcal{P}_\mathcal{R}$.

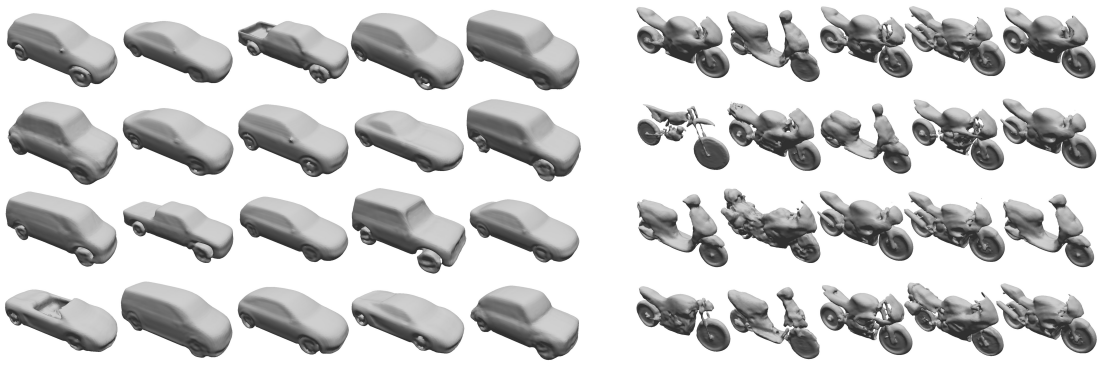

Figure 4.2: Samples from shape-space of *Car* and *Motorcycle*.

Allocentric vs. Egocentric? Object orientation can be egocentric (orientation w.r.t. camera), or allocentric (orientation w.r.t. object). Since orientation is predicted on top of RoI feature-map (generated by cropping features on a box centered on the object), it is better to choose an object-centric (allocentric) representation for learning. We illustrate this with help of Fig.4.3. Consider a car moving across the image from right to left in a straight line perpendicular to the camera axis. The azimuth of the car w.r.t. camera (egocentric), does not change, but the appearance of the cropped RoI around the car changes significantly as it moves from right side of the image to left. Whereas, objects with similar allocentric orientation also have similar appearance. So allocentric representation is equivariant w.r.t. to RoI image appearance, and is a better learnable representation. We represent object orientation in terms of viewpoint, an allocentric representation. Viewpoint describes the relative camera orientation angles $\boldsymbol{v} = [\theta, \phi, \psi]$ with the camera always looking towards the center of the object (Fig.4.3(c)). θ, ϕ, ψ denotes the azimuth, elevation, and tilt angles.

Object Position: Directly estimating the object 3D position from a cropped and resized RoI features is fundamentally an ill-posed problem. We humans are only able to estimate depth from single image when the object is of known type, and is placed in context of a bigger background. For this reason, we also do not task our network to directly estimate the depth or 3D position of the object. We instead ask our network to estimate the 2D projection of the canonical object center $\boldsymbol{c} = [x_c, y_c, 1]$, and the 2D amodal bounding box

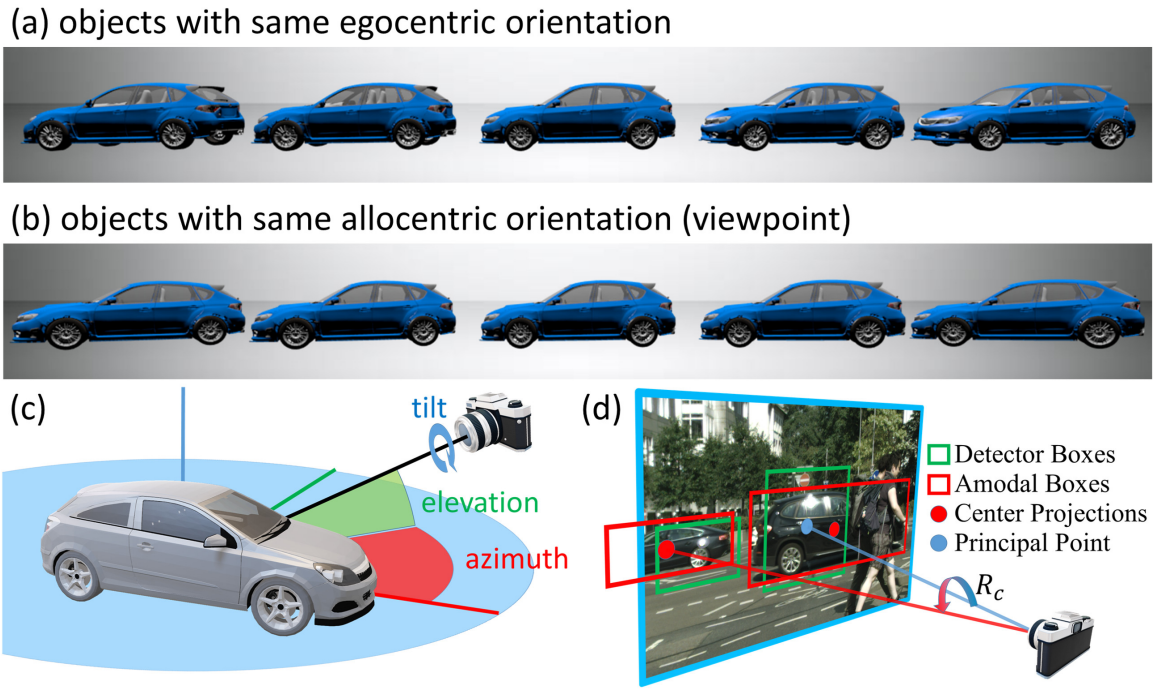

Figure 4.3: In (a) all cars in the image are at same egocentric orientation w.r.t. camera, but yet there is a significant appearance change. So if we use egocentric representation, we are asking the network to predict same angle for different image appearance. In (b) all cars in the image have same allocentric orientation, and we do not see any appearance change. Thus allocentric orientation is a better representation for learning object orientation. In (c) and (d), we illustrate the pose representation used in this paper (see § 4.4.2).

of the object $\boldsymbol{a} = [x_a, y_a, w_a, h_a]$ where (x_a, y_a) is the center of the box and (w_a, h_a) denotes the size of the box. These entities are better learnable concepts, and ground-truth data is easy to obtain [123] or already available on real-world datasets like KITTI [25] and Pascal3D+ [26].

Getting back Egocentric Pose: Given object viewpoint estimate $\boldsymbol{v}$, 2D projection of the object center $\boldsymbol{c}$ on the image, an amodal box $\boldsymbol{a}$ around the object, and camera intrinsics K_c, we can easily obtain the egocentric 3D object pose $\mathcal{P}_E \in$ **SE(3)** w.r.t. to camera. We fist compute the rotation $R_c \in$ **SO(3)**, between the camera principal axis $[0, 0, 1]^T$ and ray through object center projection $K_c^{-1}\boldsymbol{c}$. So $R_c = \Psi([0, 0, 1]^T, K_c^{-1}\boldsymbol{c})$, where the function $\Psi(\boldsymbol{p}, \boldsymbol{q})$ computes the rotation that takes vector $\boldsymbol{p}$ to align with vector $\boldsymbol{q}$: $\Psi(\boldsymbol{p}, \boldsymbol{q}) = I + [\boldsymbol{r}]_\times + [\boldsymbol{r}]_\times^2 / (1 + \boldsymbol{p} \cdot \boldsymbol{q})$, where $\boldsymbol{r} = \boldsymbol{p} \times \boldsymbol{q}$. We denote $R_v \in$ **SO(3)** as the rotation

matrix form of the viewpoint $\boldsymbol{v}$. Object center distance from camera d is computed such that the resulting shape projection tightly fits the amodal box $\boldsymbol{a}$. Then object pose $\mathcal{P}_E$ w.r.t. camera is given by:

$$\mathcal{P}_E = \begin{bmatrix} R & \boldsymbol{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \text{ where } R = R_c R_v \ , \ t = R_c[0, 0, d]^T$$

4.5 3D-RCNN Architecture

Our method adopts the Faster-RCNN/Network-on-Convolution meta-architecture [124, 125, 126]. The network comprises of a shared backbone feature extractor for the full-image, followed by region-wise sub-networks (heads) that predicts 3D shape and 3D pose in addition to traditional 2D box and class label. Figure 4.1 provides an overview.

4.5.1 Striving for 3D Equivariance

As with any Fast-RCNN++ systems, features from a RoI of arbitrary size and location $\boldsymbol{r} = [x_r, y_r, w_r, h_r]$ is extracted from the shared feature-map and then resized to a fixed resolution $f_w \times f_h$ (typically 14×14). The fixed size of RoI features allows FC layers on top of the RoI features, to share weights in-between different RoIs performing the same task. RoI feature extraction methods like RoI-Pool [126] or RoI-Align [127], transform the original feature-map with a 2D transformation to bring them to a fixed size. This 2D transformation makes it necessary, for the targets (*e.g.* 2D detection box targets) to be normalized w.r.t. RoI box. Once we have a prediction of the target by the network, they are un-normalized back for final output. The same is true for targets like 2D instance segmentation [90]. So for the 2D targets amodal-box and center-proj in our network, we

normalize them w.r.t. to RoI box $\boldsymbol{r}$ similar to [124, 126]:

$$\text{amodal-box } \hat{\boldsymbol{a}} = \left[\frac{x_a - x_r}{w_r}, \frac{y_a - y_r}{h_r}, \log\frac{w_a}{w_r}, \log\frac{h_a}{h_r}\right]$$
$$\text{center-proj } \hat{\boldsymbol{c}} = \left[\frac{x_c - x_r}{w_r}, \frac{y_c - y_r}{h_r}\right]$$

But, such 2D normalization is not possible for 3D targets like shape and pose. This is problematic and destroys equivariance, which is important for the task of shape and pose estimation. We illustrate this in Fig.4.4. Our solution to this problem is to provide the underlying 2D transformation information to the classifiers for shape and pose prediction, so that it can undo this 2D transformation.

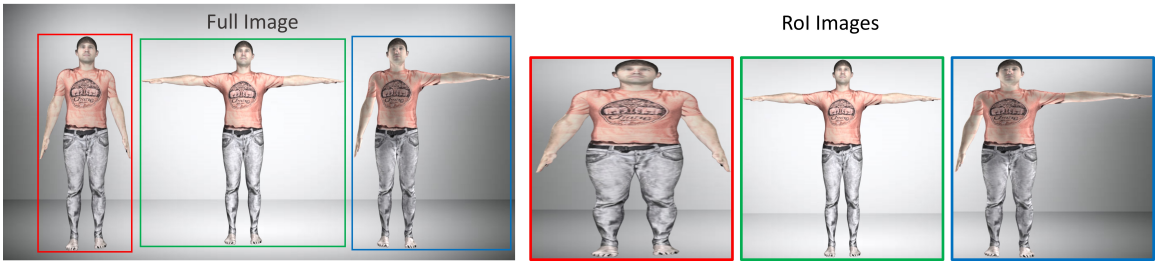

Figure 4.4: All three persons in left image have the exact same shape. In right, we show the corresponding RoI transformations when done on the raw image. Since normalization of 3D parameters w.r.t. RoI is not possible, simply training the network to predict same shape from these RoI features is sub-optimal.

We interpret the RoI crop and resize process, as an image formed by secondary RoI camera, that is rotated from the original full-image camera to look directly at a object, and having different intrinsics (zoomed-in with aspect-ratio change). Assuming known full-image camera intrinsics K_c, we compute the RoI camera intrinsics K_r as:

$$K_c = \begin{bmatrix} f_x & 0 & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix}, K_r = \begin{bmatrix} f_x f_w / r_w & 0 & f_w/2 \\ 0 & f_y f_h / r_h & f_h/2 \\ 0 & 0 & 1 \end{bmatrix}$$

The rotation between the full-image camera and RoI camera R_c, is computed in same way as described in § 4.4.2, using the prediction of object center projection center-proj. The

two cameras K_c and K_r, under pure rotation R_c, is related by the infinite homography matrix [128], $H_\infty = K_r R_c^{-1} K_c^{-1}$. H_∞ captures the 2D transformation done by RoI pooling layer, in addition to perspective distortion due to the original camera not directly looking at the center of the RoI. We use the idea of dynamic filters [129], to feed the RoI transformation information H_∞^{-1}. We use a 1×1 convolution filter on the original RoI feature-map, whose parameters are computed dynamically using a FC layer from the 9 parameters of H_∞^{-1}. We denote this additional filter as H_∞ conv (see Fig.4.1). The shape and pose classifiers use the output of H_∞ conv layer. With this additional information of H_∞^{-1} they have a better chance of learning the 3D shape and pose targets. The shape and pose targets, that our network learns are the original 3D shape and pose parameters $[\boldsymbol{v}^T, \boldsymbol{j}^T]^T$.

4.5.2 Direct 3D supervision

While it is possible to just use continuous regression loss for pose and shape, classification loss obtained by first discretizing the output-space into bins performs much better [98, 93]. Classification over-parametrizes the problem, and thus allows the network more flexibility to learn the task. It also naturally allows us to bound the range of outputs. Pose angles need to be bounded in $[-\pi, \pi]$ and each shape parameters are bounded to $[-3\sigma, 3\sigma]$. However one disadvantage of classification is that the accuracy is limited to the discretization granularity, set by the finite number of bins used. We take best of the both by combining classification and regression loss. We first perform $\mathrm{soft\,arg\,max}$ with an additional temperature T on activations of the FC layer. We then have a cross-entropy classification loss, and L1 regression loss over expectation of the $\mathrm{soft\,arg\,max}$ probabilities.

Assuming b bins for each shape parameter $\beta \in \boldsymbol{\beta}$, and $\tilde{\beta}_p$ to be the center of p-th bin, we compute β as

$$\beta = \sum_{p=1}^{b} P_\beta^p \tilde{\beta}_p, \quad P_\beta^p = \frac{\exp(\mathrm{FC}_{\mathsf{shape}}^p / T_{\mathsf{shape}})}{\sum_{q=1}^{b} \exp(\mathrm{FC}_{\mathsf{shape}}^q / T_{\mathsf{shape}})} \tag{4.1}$$

where P_β is the result of applying $\mathrm{soft}\arg\max$ with temperature T_{shape} on activations of FC_{shape}.

Since pose targets are actually angles which are periodic, we have to instead take the complex expectation. Thus each angle estimate $\theta \in [\boldsymbol{v}^T, \boldsymbol{j}^T]^T$ is computed as

$$\theta = \arg\left(\sum_{p=1}^{b} P_\theta^p e^{i\tilde{\theta}_p}\right), \quad \tilde{\theta}_p = 2\pi\frac{p-0.5}{b} - \pi \tag{4.2}$$

where P_θ like before is the result of applying $\mathrm{soft}\max$ with temperature on activations of FC_{pose}. $\tilde{\theta}_p$ is the center of the p-th bin.

For both the shape and pose targets, we combine a cross-entropy loss on the $\mathrm{soft}\max$ output, along with L1 loss on the continuous output after expectation:

$$L_{\text{shape}} = -\log(P_\beta^*) + \|\beta - \beta^*\|_{L1} \tag{4.3}$$

$$L_{\text{pose}} = -\log(P_\theta^*) + \|\theta - \theta^*\|_{L1} \tag{4.4}$$

where β^* and θ^* are the continuous ground-truth shape and pose parameters, and P_β^* and P_θ^* are the corresponding $\mathrm{soft}\max$ probabilities for the ground-truth bin.

Note that center-proj and amodal-bbx targets, are not required to be bounded like angles. Also these targets are normalized w.r.t. RoI, which has already gone through a discretization process via anchors [124] in the detection module. So we simply use L1 loss fo these two 2D targets:

$$L_{\text{center-proj}} = \|\hat{\boldsymbol{c}} - \hat{\boldsymbol{c}}^*\|_{L1}, \; L_{\text{amodal-bbx}} = \|\hat{\boldsymbol{a}} - \hat{\boldsymbol{a}}^*\|_{L1}$$

Equations (4.1) and (4.2) can also be interpreted as $\mathrm{soft}\arg\max$, and it approaches $\arg\max$ as $T \to 0$. We initialize the temperature parameters at 0.5 during training. An $\arg\max$ estimate of shape and pose instead of $\mathrm{soft}\arg\max$ would have prevented us to back-propagate gradients from the Render-and-Compare layer, which is on top of shape

and pose parameters. Our loss formulation is different from that of [93], which combines classification loss along with regression of orientation offset, thus requiring additional FC layers on top of classification FC layers. Our formulation avoids non-differentiable operations like $\arg\max$, and only introduces a scalar $\text{soft}\arg\max$ temperature parameter, which is much less than parameter-heavy FC layers.

4.5.3 Render-and-Compare Loss

Once we have a compact 3D representation of the object, it can be readily rendered from known camera calibration, and compared with 2D annotations like instance segmentation, depth-map. This allows the network to obtain supervision from more easily obtainable 2D ground-truth data.

For each RoI, we have ground-truth 2D segmentation mask G_s and/or 2D depth-map G_d. From the 3D shape and pose prediction of each RoI, we render the corresponding segmentation mask R_s, and depth-map R_d. In addition we have known binary ignore masks I_s and I_d, which have value of one at pixels which does not contribute to loss. This is useful to ignore pixels with no label, being occluded, or with undefined depth value. In its generic form Render-And-Compare loss measures the discrepancy between the rendered and ground-truth image:

$$L_{\text{render-and-compare}} = d_J(R_s, G_s; I_s) + d_{L2}(R_d, G_d, ; I_d)$$

where $d_J = 1 - J(R_s, G_s; I_s)$ is the Jaccard distance, complementary to the Jaccard index (segmentation IoU) $J(R_s, G_s; I_s)$ between R_s and G_s.

However standard 3D rendering is not differentiable. We use numerical derivatives to approximate the gradient. This is feasible since non-photorealistic rendering is fast with GPUs ($\sim$10k FPS), and dimensionality of our 3D object representation (§ 4.4) is rather small. There exists other schemes like OpenDR [116], SPSA [130] but we found simple

central derivatives to be effective and fast, since we avoid all CPU-GPU memory transfer by making use of CUDA-OpenGL interop functionality available in all recent GPUs. When using TSDF shape space, we use volume ray-casting, while SMPL shapes are rendered with traditional mesh rendering.

Render-and-Compare loss does not introduce any new learn-able parameters, but provides a joint structured loss over all the shape and pose parameters of an object. Since shape and pose representation are low dimensional and they can be readily rendered, computing gradients using numerical derivatives is feasible. Compactness (low dimensionality) of object representation, and fast rendering are desirable properties in itself.

4.5.4 Training and Inference

Joint Multi-task Loss Function: The final joint loss objective L_{joint}, that our network minimizes is the combination of losses of all the prediction targets = {shape, pose, center-proj, amodal-bbx, render-and-compare}. So, $L_{\text{joint}} = \sum_{\tau \in \text{targets}} \lambda_\tau L_\tau$ where the hyper-parameters λ_τ balances individual loss terms. Depending on the data-sample, certain loss terms will be unavailable. For example we do not have ground-truth shape target for real-world data-samples.

Training: Starting with Imagenet [131] pre-training, we first train our network on the synthetic images rendered from CAD models similar to [100, 132]. Unlike [100, 132], we render multiple objects per-image and we use roughly 20K synthetic images per class, compared to a million images per class as in [100, 132]. See Appendix A) for more information on synthetic data generation process. After this bootstrapping, we then fine-tune the network on KITTI and PASCAL datasets for our experiments along with Render-and-Compare loss, whenever such data is available. We use SGD for all our experiments, and the network is trained end-to-end.

Inference: Our inference step is efficient and only involves a feed forward pass through network, without any post-processing or costly test time optimization steps. With ResNet-50 like backbone, our method produces full 3D shape and pose of all objects in an image in under 200ms. Note that this is more that 30x faster than methods like [99, 100] even without considering object detection time, while they still provide only 3D orientation output.

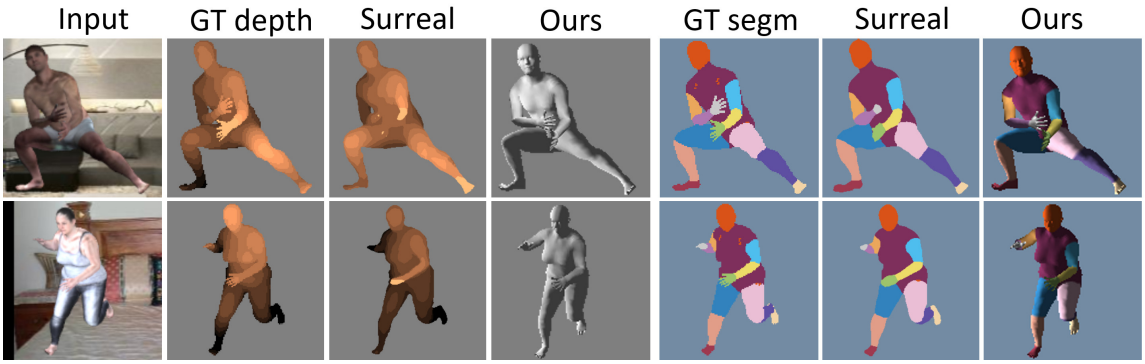

Figure 4.5: Qualitative comparison of our approach with [132] on recently released SURREAL [132] dataset. Note that [132] trains two distinct conv-nets specific to the task of depth prediction and body parts segmentation. Our method predicts the 3D shape and pose of each body. Depth and body pats segmentation are generated by simply rendering the predicted output shape from camera view.

Figure 4.6: Qualitative demonstration of our approach working on KITTI [25] dataset. Input images are shown in first column, and the corresponding 3D object pose and shape output are shown in second column. Each object instance has been colored randomly. Third column shows the projection of the 3D object instance reconstructions on the input image which demonstrates the capability of producing accurate 2D instance segmentation, which comes for free due our holistic 3D representation.

Method	Bicycle				Motorcycle				Car			
	AVP_4	AVP_8	AVP_{16}	AVP_{24}	AVP_4	AVP_8	AVP_{16}	AVP_{24}	AVP_4	AVP_8	AVP_{16}	AVP_{24}
Pepik *et al.* [133]	43.9	40.3	22.9	16.7	31.8	32.0	16.7	10.5	36.9	36.6	29.6	24.6
Tulsiani & Malik [99]	59.4	54.8	42.0	33.4	61.1	59.5	38.8	34.3	55.2	51.5	42.8	40.0
RenderForCNN [100]	50.5	41.1	25.8	22.0	50.8	39.9	31.4	24.4	41.8	36.6	29.7	25.5
Poirson *et al.* [97]	62.1	56.4	39.6	29.4	62.7	58.6	40.4	30.3	51.4	45.2	35.4	35.7
Massa *et al.* [98]	67.0	62.5	43.0	39.4	71.5	64.0	49.4	37.5	58.3	55.7	46.3	44.2
Xiang *et al.* [134]	60.4	36.3	23.7	16.4	60.7	37.0	23.4	19.9	48.7	37.2	31.4	24.6
Our method	**74.3**	**67.2**	**51.0**	**42.1**	**74.4**	**72.3**	**52.2**	**47.1**	**71.8**	**65.5**	**55.6**	**52.1**

Table 4.1: Joint detection and viewpoint evaluation on Pascal3D+ dataset [26] for *Bicycle, Motorcycle*, and *Car* category.

Method	Easy			Moderate			Hard		
	$AP\uparrow$	$AOS\uparrow$	$AAE\downarrow$	$AP\uparrow$	$AOS\uparrow$	$AAE\downarrow$	$AP\uparrow$	$AOS\uparrow$	$AAE\downarrow$
SubCNN [134]	90.53%	85.90%	12.24°	85.71%	84.21%	15.20°	72.71%	70.61%	17.14°
Ours (orginal box)	90.53%	90.50%	1.99°	85.71%	85.57%	**4.51°**	72.71%	71.98%	**6.50°**
Ours (rendered box)	**90.76%**	**90.73%**	**1.98°**	**89.31%**	**89.15%**	4.90°	**79.89%**	**79.51%**	7.94°

Table 4.2: Evaluation against [134] using the same 2D object detector as input on KITTI train/validation split of [101]. Notice the big improvement in object detection AP when using rendered box.

Method	Easy			Moderate			Hard		
	$AP\uparrow$	$AOS\uparrow$	$AAE\downarrow$	$AP\uparrow$	$AOS\uparrow$	$AAE\downarrow$	$AP\uparrow$	$AOS\uparrow$	$AAE\downarrow$
Deep3DBox [93]	97.77%	97.52%	5.75°	96.85%	96.34%	8.30°	81.08%	80.44%	10.25°
Ours (orginal box)	97.77%	97.69%	3.15°	96.85%	96.60%	5.78°	81.08%	80.79%	6.88°
Ours (rendered box)	97.76%	97.69%	3.15°	96.79%	96.55%	5.62°	80.98%	80.69%	6.84°

Table 4.3: Evaluation with [93] using the same initial 2D detections provided by the authors. Notice that our orientation estimate is much more accurate. Since the author provided detections have been trained on additional data, we do not see much improvement by using rendered box.

Method	Bicycle		Motorcycle		Car	
	$Acc_{\pi/6}\uparrow$	$MedErr\downarrow$	$Acc_{\pi/6}\uparrow$	$MedErr\downarrow$	$Acc_{\pi/6}\uparrow$	$MedErr\downarrow$
Tulsiani & Malik [99]	0.77	17.7°	0.88	14.7°	0.89	9.1°
RenderForCNN [100]	0.83	14.8°	0.78	16.7°	0.88	6.0°
Deep3DBox [93]	0.83	12.5°	0.86	12.3°	0.90	5.8°
Our method	**0.88**	**8.4°**	**0.95**	**6.8°**	**0.96**	**3.0°**

Table 4.4: Viewpoint estimation with ground-truth detections on Pascal3D+ [26]. $MedErr$ is median angular error (lower is better) in degrees, while $Acc_{\pi/6}$ measures viewpoint accuracy (higher is better) with $\pi/6$ as threshold. See main paper for more details.

Method	Easy			Moderate			Hard		
	$AP\uparrow$	$AOS\uparrow$	$AAE\downarrow$	$AP\uparrow$	$AOS\uparrow$	$AAE\downarrow$	$AP\uparrow$	$AOS\uparrow$	$AAE\downarrow$
3DOP [135]	93.04%	91.44%	15.07°	88.64%	86.10%	19.49°	79.10%	76.52%	20.81°
Mono3D [136]	92.33%	91.01%	13.73°	88.66%	86.62%	17.45°	78.96%	76.84%	18.86°
SubCNN [134]	90.81%	90.67%	4.50°	89.04%	88.62%	7.88°	79.27%	78.68%	9.90°
Deep3DBox [93]	92.98%	92.90%	3.36°	89.04%	88.75%	6.54°	77.17%	76.76%	8.36°
DeepMANTA [94]	**97.25%**	**97.19%**	2.85°	**90.03%**	**89.86%**	4.98°	**80.62%**	**80.39%**	6.12°
Our Method	90.02%	89.98%	**2.42°**	89.39%	89.25%	**4.54°**	80.29%	80.07%	**6.00°**

Table 4.5: Joint detection and orientation evaluation on KITTI test split for all difficulty levels. Apart from AP and AOS, we also report Average Angular Error (AAE). AAE (lower is better) gives a measure of average angular error in orientation normalized by the detector precision and is thus a better metric to study the performance of orientation prediction. Our method has the lowest AAE for all cases.

Method	Cars		Person	
	$MedErr\downarrow$	$IoU\uparrow$	$MedErr\downarrow$	$IoU\uparrow$
Ours full	1.60°	91.9	4.13°	69.1
w/o Render-and-Compare	1.75°	86.8	5.72°	65.3
w/o H_∞ conv layer	1.73°	89.5	4.94°	66.7

Table 4.6: Ablation study on our synthetic dataset (see § 4.6.3). We evaluate viewpoint estimation error and segmentation IoU score.

4.6 Experiments

We focus our experiments on the two most common object categories in urban scene: *Car* and *Person*. These two object classes also covers both rigid (*Car*) and articulated (*Person*) objects and thus demonstrates our method's applicability to diverse shape and pose models.

We benchmark our method on challenging PASCAL3D+ [26] and KITTI [25] dataset. We benchmark of our model on PASCAL3D+ and KITTI dataset for joint detection and pose estimation. We also provide detailed ablation study for 3D pose estimation. Our method achieves superior performance on both PASCAL3D+ [26] and KITTI [25] datasets, and outperformed all recent methods by a significant margin. We also present ablation study of segmentation IoU and viewpoint estimation error in our synthetic dataset (see § 4.5.4) for *Person* and *Car* objects. Fig.4.5 shows qualitative results on [132], by training our pipeline for *Person* using our synthetic dataset. See additional results in supplementary.

4.6.1 Analysis on Pascal3D+ dataset

We first evaluate our method on the primary PASCAL3D+ task of joint detection and viewpoint estimation. We report results using Average Viewpoint Precision (AVP) under different quantization of the angles, as proposed by [26]. Our results are listed in Table 4.1. Our system improves upon all previous methods by at least 10% over all quantizations.

To better understand the efficacy of pose estimation of our network, we follow [99, 100, 93] to evaluate viewpoint on ground-truth boxes. Evaluating viewpoint prediction on ground-truth boxes, provides an upper-bound of viewpoint accuracy independent of the object detector used. The viewpoint estimation error is measured as geodesic distance over the rotation group **SO(3)**. We report $Acc_{\pi/6}$ which measures accuracy thresholded at $\frac{\pi}{6}$ and the median angular error $MedErr$. This is the same evaluation metric originally used in [99] and then in [100] and [93]. Please refer to [100] or [99] for more details. Our results are summarized in Table 4.4. Our method improves $Acc_{\pi/6}$ by 5 points over the previous best, and median angular error is reduced by $\sim 50\%$ from 5.8° to 3.0°. We also experimented with VGG16 [48] as our backbone and got similar improvements.

4.6.2 Analysis on KITTI dataset

In this section, we evaluate our method on KITTI object detection and orientation benchmark [25]. We envision our system for autonomous driving applications. So KITTI is a good test-bed as it involves many challenges of real-world urban driving. Qualitative results on KITTI dataset are shown in Fig.4.6. We adopt the official evaluation metric of Average Precision (AP) for detection and Average Orientation Similarity (AOS) for joint detection and pose estimation. We also report Average Angular Error (AAE) defined as $\arccos(2 * (AOS/AP) - 1)$ which gives a detection normalized measure of average orientation error.

Results on the KITTI test set is shown in Table 4.5. Since test set labels are not publicly available, we follow [134, 93] to divide the official training set into disjoint training and

validation set for our ablation study. Fo the ablation study, we use the same detection input as Deep3DBox [93] and SubCNN [134] provided by the authors. Our methods beats [93, 134] on both AOS and AAE metrics. Instead of just using the input detector boxes as final box output, we can also generate 2D detection box by simply rendering our output 3D scene representation. This rendered box significantly improves the detection performance AP over the input detector of [134]. The results are summarized in Table 4.2 and Table 4.3.

4.6.3 Ablation on Synthetic dataset

As mentioned in § 4.5.4, we have generated synthetic images rendered using the models of [100, 132] to [100, 132], but with multiple objects per image to improve training efficiency (see Appendix A). We use this data to do an ablation study of Render-and-Compare loss (§ 4.5.3) and H_∞ conv layer (§ 4.5.1). For our ablation study, we split this synthetic data into train (80%), val (10%) and test(10%) sets. We study the effect of removing either Render-and-Compare loss or H_∞ conv layer from our full pipeline. The results are summarized in Table 4.6. Even though, the synthetic data has ground-truth for all our targets, removing Render-and-Compare has a significant impact. This is because Render-and-Compare loss is directly maximizing the IoU and both shape and pose needs to be predicted accurately for perfect segmentation. Thus, Render-and-Compare provides a joint structured loss over all the shape and pose parameters.

4.7 Conclusion

We present a fast inverse-graphics approach for 3D scene understanding from images. Our network estimates the full 3D shape and pose of each object instance in an image. This rich 3D representation brings several advantages: (a) traditional vision outputs like 2D detection, segmentation, and depth-maps comes free and; (b) allows the network to be also trained with 2D supervision. We present novel representation of shape and pose, that strives towards better 3D equivariance and helps the deep model to learn the mapping from input

image region to full 3D shape and pose. We evaluate on challenging real-world datasets of Pascal3D+ and KITTI where our method achieves state-of-the-art results in multiple tasks. Our work is suitable for several real-world applications like autonomous driving.

5

CONCLUSION

In this thesis, we present approaches to solve different aspects of the problem of scene understanding from 2D images. We present three scene understanding frameworks with increasing richness in representation. We also demonstrate seamless integration of different constraints and prior knowledge into our model and an effective fusion of multiple measurements into a final prediction. Our work is suitable for several real-world applications like autonomous driving, game level building, cartography, *etc*.

In Chapter 2, we present a structured prediction model for semantic segmentation that reasons jointly over *all* pixels in a video–not just individual frames. As a result, we obtain an improved and temporally consistent video semantic segmentation. The key observation of this work is that a straight-forward temporal extension of 2D image-space regularization can be detrimental for videos as they do not capture temporal correspondence information. We propose to solve for a low-dimensional feature space embedding that captures correspondence information between pixels, and is thus more effective for pairwise regularization, while still operating on the pixels lying in the 2D image space.

In Chapter 3, we present a semantic 3D reconstruction framework that jointly builds a volumetric reconstruction of the scene and assigns a semantic label to each voxel. We formulate the problem as a 3D spatial CRF where each voxel is globally assigned a discrete semantic label indicating whether it is a free voxel or belongs to a particular solid semantic object category. We also introduce several new factors to account for sparse input 3D reconstruction and ambiguities in semantic labelling. Since the pairwise smoothness terms are in true 3D space, correspondence information is implicitly built into the representation, unlike the 2D video segmentation work in Chapter 2. However, unlike Chapter 2, this

system can only represent a static scene.

Both Chapter 2 and Chapter 3 provide a semantic understanding of the scene but lack the concept of object instances. Therefore, questions such as, "How many *Cars* in the scene?" remain unanswered. In Chapter 4, we present a framework for instance-level 3D scene understanding. We adopt an inverse-graphics approach for this problem and present a unified deep convolutional network named 3D-RCNN that learns to map image regions to full 3D shape and pose of all object instances in the image. Many traditional 2D vision tasks, like instance segmentation and depth-maps, can be obtained by simply rendering the output 3D scene model. We present novel representations of shape and pose, that strive towards better 3D equivariance and generalization. In order to exploit the rich amount of supervisory signals in the form of 2D annotations, like segmentation, our proposed 3D-RCNN framework also includes a differentiable Render-and-Compare loss module that allows 3D shape and pose to be learned with 2D supervision.

5.1 Key Take-Away Points

The following are the key "take-away" points of this thesis.

Representation choice: Representation choice is a critical component when designing systems for scene understanding. We need representations and tasks that make the AI agent to *study harder*. Contrary to a purely data-driven black-box representation which is learned end-to-end for a specific task, we argue for generic, explicit 3D representation. Asking a neural network to predict the full 3D shape and pose of an object is a much harder task than simple semantic classification tasks commonly practiced in computer vision. By asking the network to predict the complete 3D shape and pose parameters helps the learning process to avoid shortcuts and generalize better. Constraints like spatial smoothness, motion smoothness, shape constancy are better expressed with an explicit 3D representation. However, most of the observations and training data are available in 2D image domain. 3D

representation with differentiable rendering capability offers the best of both worlds, as the 3D representation can be readily rendered and compared with 2D observations.

Rich Semantic 3D representation: In this thesis, we propose a rich 3D scene model for complex dynamic scenes. With our object based representation, we can have both class-specific and instance-specific semantic priors. The proposed framework uses a detailed 3D scene representation and reasons about interactions between scene elements in 3D space. A compact 3D representation makes our method more suitable and efficient for real-world 3D applications. The output from our system can be directly used for tasks like path-planning, augmented-reality, or to accurately predict an object's 3D location in the future. The predictive capability of the 3D scene representation also paves the way for semi-supervised predictive learning. Another major benefit of scene understanding with a rich 3D scene model is that traditional 2D scene representations like semantic segmentation, instance segmentation, and 2D depth maps are available for free. They can be generated by simply rendering the output 3D scene model. In contrast, traditional scene understanding methods are designed specifically for only a single task, like instance segmentation or depth map.

Inverse graphics: *Analysis-by-synthesis* with a 3D scene model is like "solving *vision* as *inverse-graphics*." Inverse graphics is a conceptually elegant way for solving vision problems. Although it has a long history, inverse graphics has thus far only been successful for a very limited set of problems. However, there has recently been a re-emergence of interest in inverse graphics, mainly due to the possibility of using powerful deep learning methods as a bottom-up process to cut down on the search space. Most of these approaches are still restricted to very simple scenes. In § 4.1, we highlight some of the key design choices that enables us to scale an inverse graphics approach for complex 3D scene understanding. For example, it is important to have a unified framework that allows both the bottom-up features and the top-down analysis to be trained end to end. This is achieved with a differentiable Render-and-Compare loss layer that also allows training from more commonly

available 2D annotations.

3D equivariance: Estimating 3D properties or entities from a 2D image region requires more careful design to preserve the equivariance properties. One of the key reason behind the success of convolutional networks is the built-in equivariance to translation. Understanding the invariance or equivariance demands of the task at hand is of fundamental importance behind "good" network architectures [92, 90, 91]. Classification tasks demand invariance, whereas segmentation or bounding box regression demands equivariance to 2D transformations. Since shape and pose are 3D entities, normalization of these parameters w.r.t. 2D Region-of-Interest (RoI) transformations is not possible in the same manner as it is done for 2D entities, like bounding box parameters and instance segmentation. Instead, we capture the 2D transformations performed by RoI pooling layers and feed them to shape and pose classifiers (see § 4.5.1).

5.2 Future Work

Automatic 3D scene understanding from images is still in its infancy, and current approaches have a long way to go before they match the effectiveness of human vision. We believe revisiting "vision as inverse-graphics" approaches, in light of the recent advances in deep neural networks, and coupled with the availability of large-scale 3D CAD models, is an important step in that direction. We present such an approach in Chapter 4 of the thesis. However, the presented approach still leaves much to be desired; in the following paragraphs, we discuss possible future works.

Joint reasoning of both *stuff* and *things*: The semantic label of an object can be categorized into *stuff* (*e.g.*, building, terrain, *etc.*) or *things* (*e.g.*, car, pedestrian, traffic-light, *etc.*) [137]. *Stuff* have no specific shape or spatial extent, whereas *things* exhibit a strong specific shape and spatial extent. In Chapter 4, we reason only about *things* which have

a countable number of instances and ignore *stuff* object categories. In contrast, the approaches presented in Chapter 2 and Chapter 3 lack the concept of object instances. Joint reasoning of both static *stuff* objects and *thing* object instances will provide a full understanding of the entire scene. One way to achieve this will be to use the 3D-RCNN framework (Chapter 4) for object instances combined with static semantic 3D reconstruction framework (Chapter 3) for *stuff*. We can additionally have a test-time inference that reasons jointly about all the objects such that likelihood of the observed 2D measurements is maximized.

Shared shape space: Different 3D shape representations are used throughout this thesis. In Chapter 3, we use a non-parametric representation of shape with voxels. In Chapter 4, we use a low-dimensional class-specific shape representation (TSDF-PCA for rigid objects and SMPL for human bodies). However, learning a separate shape-space for each category of objects may not scale to large categories of objects. Learning a shared shape-space model for all object categories has many advantages, including smaller model size and simplicity. For example, most vehicles have wheels and can thus share the representation of *wheels*. Furthermore, when the number of CAD models for a particular object category, such as *Bicycle*, is very small, a shared shape-space can exploit knowledge learned for similar categories, such as *Motorcycle*. Ideally, such shared shape-space should encode within-category shape variation. It should have a specific parameter dedicated to switch between different categories. Finally, the learned shape space model should be low dimensional and fast to decode, so as to be effectively used in differentiable Render-and-Compare loss functions.

Differentaibale tracking: The 3D-RCNN framework is designed to work with a single image input. However tracking object instances over time is useful for many applications. One possible solution is to combine the single image predictions from 3D-RCNN framework with an appropriate graphical model and incorporate additional constraints available

in a video. In a dynamic scene, an object instance has constant shape parameters, while its pose may change over time. Note that this assumption is true even for articulated objects, as we use pose-agnostic parametric representation for object shape as described in § 4.4.1. The joint angles for articulated object are part of the pose parameters, and the shape parameters define the object shape in a canonical rest pose. The task of the tracker is then to select a set of *tracklets* (object hypotheses over multiple frames) and predict the object shape and its pose over time, that best describe the observed data and priors like shape constancy. Additionally, if the tracking framework is differentiable then the whole system can be trained end-to-end with videos.

Semi-supervised predictive learning: The 3D representation we present in this thesis paves the way for semi-supervised fine-tuning from a large amount of readily-available unlabelled data. For example, it is easy to collect large amount of real-world images registered with sparse depth data from LIDARs like in KITTI dataset. The 3D-RCNN model trained on a relatively small amount of labelled data can then be fine tuned via Render-and-Compare loss on this large amount of sparse depth data. If we also have a differentiable tracking, we can even train with unlabeled videos which have known camera parameters. Since the 3D-RCNN framework predicts the full 3D structure of each object, we can use them to obtain 2D correspondences across images in the video. The 2D correspondence predictions can then be exploited for training by minimizing photometric errors. These loss functions are only possible due to the underlying 3D representation predicted by our framework, and are far more informative than traditional unsupervised loss functions. Note that the learning algorithm is still based on traditional supervised learning techniques, often known as *predictive* learning.

Appendices

APPENDIX A

SYNTHETIC DATASET

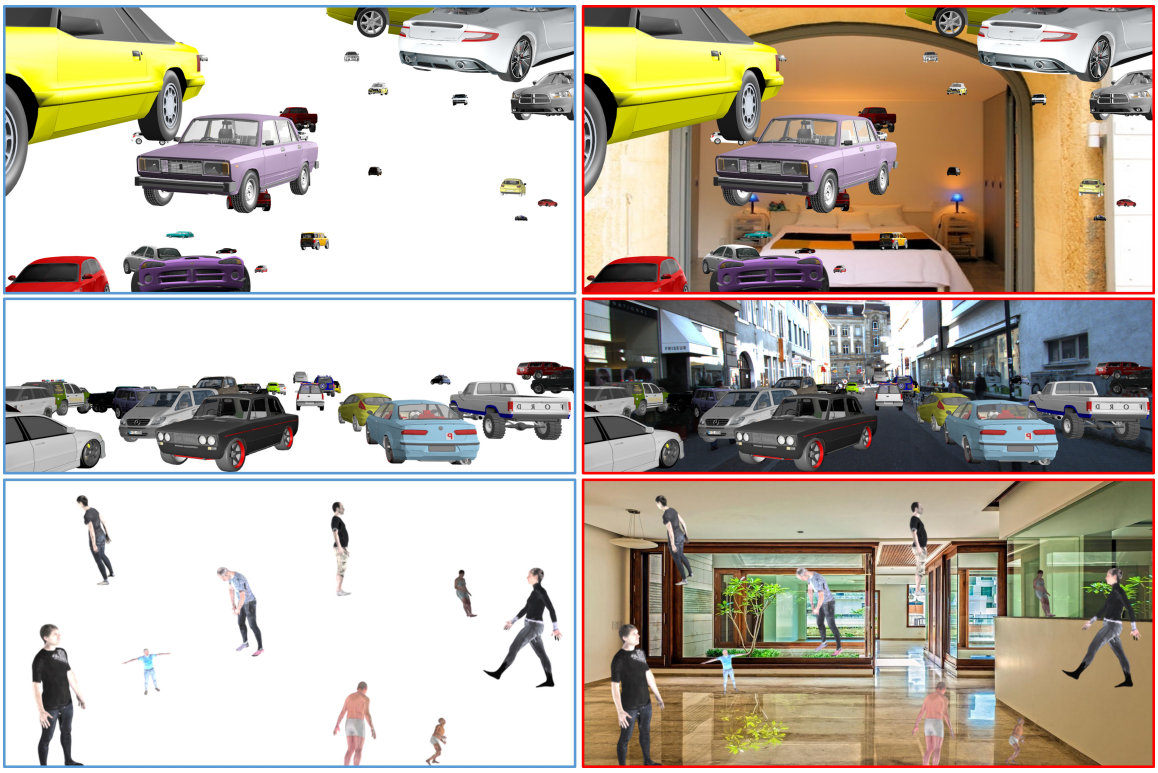

Figure A.1: Multiple objects are rendered per-image with transparent background as shown in left. We also use statistics generated from KITTI and Pascal3D+ for sampling different orientations and positions of the CAD models. We then composite these rendered images with random background.

To bootstrap the learning process of the 3D-RCNN model presented in Chapter 4, we generate synthetic images rendered from CAD models, so that we can have direct supervision for each factored targets like shape and pose. This is important as starting the learning process with only Render-and-Compare loss may fail to converge. Our synthetic data generation process is similar to [100, 132]. However, unlike [100, 132], we render multiple objects per-image and we use roughly 20K synthetic images per class, compared to million images per class as in [100, 132]. In [100, 132], only a single object is rendered per image.

So the synthetic data does not provide intra-class occlusion examples (see Fig.A.1). But more importantly, million images with only one object per image is very inefficient for training. The rendering pipeline is similar to [100, 132].

We use 3D CAD models [89, 88] for rigid objects and SMPL [115] for articulated objects. We first generate a distribution of object poses from Pascal3D and KITTI training data for rigid objects. For articulated *Person* category, we sample poses and shapes from SMPL parameters provided by [132], which are fitted to CMU motion capture dataset. Multiple objects are rendered per-image with transparent background which is then composited with a random background image to obtain the final image used for training. See Fig.A.1 for examples of images from this synthetic dataset.

REFERENCES

[1] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, *et al.*, "End to end learning for self-driving cars," *ArXiv preprint arXiv:1604.07316*, 2016.

[2] S. Shalev-Shwartz and A. Shashua, "On the sample complexity of end-to-end training vs. semantic abstraction training," *ArXiv preprint arXiv:1604.06915*, 2016.

[3] J. B. Tenenbaum, C. Kemp, T. L. Griffiths, and N. D. Goodman, "How to grow a mind: Statistics, structure, and abstraction," *Science*, vol. 331, no. 6022, pp. 1279–1285, 2011.

[4] J. Pearl, "Theoretical impediments to machine learning," 2017.

[5] H. v. Helmholtz, *Handbook of Physiological Optics*. Dover Publications, 1867.

[6] G. E. Hinton, P. Dayan, B. J. Frey, and R. M. Neal, "The wake-sleep algorithm for unsupervised neural networks," *Science*, vol. 268, no. 5214, p. 1158, 1995.

[7] P. Dayan, G. E. Hinton, R. M. Neal, and R. S. Zemel, "The helmholtz machine," *Neural computation*, vol. 7, no. 5, pp. 889–904, 1995.

[8] B. G. Baumgart, "Geometric modeling for computer vision.," PhD thesis, Stanford University, 1974.

[9] D. Kersten, P. Mamassian, and A. Yuille, "Object perception as bayesian inference," *Annu. Rev. Psychol.*, vol. 55, pp. 271–304, 2004.

[10] A. Yuille and D. Kersten, "Vision as bayesian inference: Analysis by synthesis?" *Trends in cognitive sciences*, vol. 10, no. 7, pp. 301–308, 2006.

[11] V. Nair, J. Susskind, and G. E. Hinton, "Analysis-by-synthesis by learning to invert generative black boxes," in *International Conference on Artificial Neural Networks*, Springer, 2008, pp. 971–981.

[12] C. K. Williams, M. Revow, and G. E. Hinton, "Instantiating deformable models with a neural net," *Computer vision and image understanding*, vol. 68, no. 1, pp. 120–126, 1997.

[13] I. Yildirim, T. D. Kulkarni, W. A. Freiwald, and J. B. Tenenbaum, "Efficient and robust analysis-by-synthesis in vision: A computational framework, behavioral tests,

and modeling neuronal representations," in *Annual Conference of the Cognitive Science Society*, 2015.

[14] T. Sharp, C. Keskin, D. Robertson, J. Taylor, J. Shotton, D. Kim, C. Rhemann, I. Leichter, A. Vinnikov, Y. Wei, *et al.*, "Accurate, robust, and flexible real-time hand tracking," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, ACM, 2015, pp. 3633–3642.

[15] D. Joseph Tan, T. Cashman, J. Taylor, A. Fitzgibbon, D. Tarlow, S. Khamis, S. Izadi, and J. Shotton, "Fits like a glove: rapid and reliable hand shape personalization," in *CVPR*, 2016.

[16] S. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, G. E. Hinton, *et al.*, "Attend, Infer, Repeat: fast scene understanding with generative models," in *NIPS*, 2016.

[17] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum, "Deep convolutional inverse graphics network," in *NIPS*, 2015.

[18] L. Romaszko, C. K. Williams, P. Moreno, P. Kohli, J. Czarnowski, S. Leutenegger, A. J. Davison, R. Khasanova, P. Frossard, I. Melekhov, *et al.*, "Vision-As-Inverse-Graphics: obtaining a rich 3d explanation of a scene from a single image," in *ICCV Workshop on Geometry Meets Deep Learning*, 2017.

[19] J. Valentin, A. Dai, M. Niessner, P. Kohli, P. H. Torr, S. Izadi, and C. Keskin, "Learning to navigate the energy landscape," *ArXiv preprint arXiv:1603.05772*, 2016.

[20] V. Jampani, S. Nowozin, M. Loper, and P. V. Gehler, "The informed sampler: a discriminative approach to bayesian inference in generative computer vision models," *Computer Vision and Image Understanding*, vol. 136, pp. 32–44, 2015.

[21] P. Moreno, C. K. Williams, C. Nash, and P. Kohli, "Overcoming occlusion with inverse graphics," in *ECCV 2016 Workshops*, Springer, 2016, pp. 170–185.

[22] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black, "Keep it SMPL: automatic estimation of 3d human pose and shape from a single image," in *ECCV*, 2016.

[23] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "Shapenet: an information-rich 3d model repository," Stanford University — Princeton University — Toyota Technological Institute at Chicago, Tech. Rep. arXiv:1512.03012 [cs.GR], 2015.

[24] L. Pishchulin, S. Wuhrer, T. Helten, C. Theobalt, and B. Schiele, "Building statistical shape spaces for 3d human modeling," *ArXiv preprint arXiv:1503.05860*, 2015.

[25] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *CVPR*, 2012.

[26] Y. Xiang, R. Mottaghi, and S. Savarese, "Beyond pascal: A benchmark for 3d object detection in the wild," in *WACV*, 2014.

[27] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes dataset for semantic urban scene understanding," in *CVPR*, 2016.

[28] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. Lopez, "The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *CVPR*, 2016.

[29] A Gaidon, Q Wang, Y Cabon, and E Vig, "Virtual worlds as proxy for multi-object tracking analysis," in *CVPR*, 2016.

[30] S. R. Richter, Z. Hayder, and V. Koltun, "Playing for benchmarks," in *ICCV*, 2017.

[31] A. Kundu, V. Vineet, and V. Koltun, "Feature space optimization for semantic video segmentation," in *CVPR*, 2016.

[32] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected CRFs with Gaussian edge potentials," in *NIPS*, 2011.

[33] ——, "Parameter learning and convergent inference for dense random fields," in *ICML*, 2013.

[34] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, 2015.

[35] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," in *ICLR*, 2015.

[36] G. Lin, C. Shen, A. van dan Hengel, and I. Reid, "Efficient piecewise training of deep structured models for semantic segmentation," in *CVPR*, 2016.

[37] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr, "Conditional random fields as recurrent neural networks," in *ICCV*, 2015.

[38] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognition Letters*, vol. 30, no. 2, 2009.

[39] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," *ACM Transactions on Graphics*, vol. 23, no. 3, 2004.

[40] D. Krishnan, R. Fattal, and R. Szeliski, "Efficient preconditioning of Laplacian matrices for computer graphics," *ACM Transactions on Graphics*, vol. 32, no. 4, 2013.

[41] P. Dollár and C. L. Zitnick, "Fast edge detection using structured forests," *PAMI*, vol. 37, no. 8, 2015.

[42] H. A. van der Vorst, "Bi-cgstab: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems," *SIAM Journal on Scientific and Statistical Computing*, vol. 13, no. 2, 1992.

[43] J. W. Ruge and K. Stüben, "Algebraic multigrid," in *Multigrid Methods*, SIAM, 1987.

[44] N. Komodakis, N. Paragios, and G. Tziritas, "MRF energy minimization and beyond via dual decomposition," *PAMI*, vol. 33, no. 3, 2011.

[45] J. Shotton, J. M. Winn, C. Rother, and A. Criminisi, "TextonBoost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context," *IJCV*, vol. 81, no. 1, 2009.

[46] L. Ladicky, C. Russell, P. Kohli, and P. H. Torr, "Associative hierarchical CRFs for object class image segmentation," in *ICCV*, 2009.

[47] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *ICLR*, 2016.

[48] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.

[49] T. Brox and J. Malik, "Large displacement optical flow: Descriptor matching in variational motion estimation," *PAMI*, vol. 33, no. 3, 2011.

[50] M. Menze, C. Heipke, and A. Geiger, "Discrete optimization for optical flow," in *GCPR*, 2015.

[51] N. Sundaram, T. Brox, and K. Keutzer, "Dense point trajectories by GPU-accelerated large displacement optical flow," in *ECCV*, 2010.

[52] P. Sturgess, K. Alahari, L. Ladicky, and P. H. Torr, "Combining appearance and structure from motion features for road scene understanding," in *BMVC*, 2009.

[53] M. Everingham, L. J. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *IJCV*, vol. 88, no. 2, 2010.

[54] J. Tighe and S. Lazebnik, "Superparsing – scalable nonparametric image parsing with superpixels," *IJCV*, vol. 101, no. 2, 2013.

[55] S. Tripathi, S. Belongie, Y. Hwang, and T. Q. Nguyen, "Semantic video segmentation: Exploring inference efficiency," in *ISOCC*, 2015.

[56] B. Liu and X. He, "Multiclass semantic video segmentation with object-level active inference," in *CVPR*, 2015.

[57] V. Badrinarayanan, A. Handa, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling," *ArXiv:1505.07293*, 2015.

[58] B. Liu, X. He, and S. Gould, "Multi-class semantic video segmentation with exemplar-based object reasoning," in *WACV*, 2015.

[59] O. Miksik, D. Munoz, J. A. Bagnell, and M. Hebert, "Efficient temporal consistency for streaming video scene analysis," in *ICRA*, 2013.

[60] Q. Chen and V. Koltun, "Full flow: Optical flow estimation by global optimization over regular grids," in *CVPR*, 2016.

[61] M. Rubinstein, C. Liu, and W. T. Freeman, "Towards longer long-range motion trajectories," in *BMVC*, 2012.

[62] A. Kundu, Y. Li, F. Dellaert, F. Li, and J. M. Rehg, "Joint semantic segmentation and 3d reconstruction from monocular video," in *ECCV*, 2014.

[63] C. Häne, C. Zach, A. Cohen, R. Angst, and M. Pollefeys, "Joint 3d scene reconstruction and class segmentation," in *CVPR*, 2013.

[64] S. Sengupta, E. Greveson, A. Shahrokni, and P. H. Torr, "Urban 3d semantic modelling using stereo vision," in *ICRA*, 2013.

[65] M. Jancosek and T. Pajdla, "Multi-view reconstruction preserving weakly-supported surfaces," in *CVPR*, 2011.

[66] G. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: a high-definition ground truth database," *PRL*, vol. 30, no. 2, pp. 88–97, 2009.

[67] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and recognition using structure from motion point clouds," in *ECCV*, 2008.

[68] L. Ladicky, P. Sturgess, C. Russell, S. Sengupta, Y. Bastanlar, W. Clocksin, and P. H. Torr, "Joint optimisation for object class segmentation and dense stereo reconstruction," in *BMVC*, 2010.

[69] G. Floros and B. Leibe, "Joint 2d-3d temporally consistent segmentation of street scenes," in *CVPR*, 2012.

[70] B. Liu, S. Gould, and D. Koller, "Single image depth estimation from predicted semantic labels," in *CVPR*, 2010.

[71] D. Hoiem, A. Efros, and M. Hebert, "Recovering surface layout from an image," *IJCV*, vol. 75, no. 1, pp. 151–172, 2007.

[72] A. Saxena, S. Chung, and A. Ng, "3-d depth reconstruction from a single still image," *IJCV*, vol. 76, no. 1, pp. 53–69, 2008.

[73] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *ICML*, 2001.

[74] C. Sutton and A. McCallum, "An introduction to conditional random fields," *PAMI*, vol. 4, no. 4, pp. 267–373, 2012.

[75] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics.* MIT Press, 2005.

[76] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, 2009.

[77] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3D mapping framework based on octrees," *Autonomous Robots*, 2013.

[78] S. Liu and D. B. Cooper, "Ray markov random fields for image-based 3d modeling: Model and efficient inference," in *CVPR*, 2010.

[79] P. Kohli, L. Ladick, and P. H. Torr, "Robust higher order potentials for enforcing label consistency," *IJCV*, vol. 82, no. 3, pp. 302–324, 2009.

[80] N. Komodakis and N. Paragios, "Beyond pairwise energies: Efficient optimization for higher-order mrfs," in *CVPR*, 2009.

[81] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert, "Isam2: Incremental smoothing and mapping using the Bayes tree," *IJRR*, vol. 31, pp. 217–236, 2012.

[82] S. Agarwal, K. Mierle, and Others, *Ceres solver*, `https://code.google.com/p/ceres-solver/`, 2012.

[83] J. H. Kappes, M. Speth, G. Reinelt, and C. Schnorr, "Towards efficient and exact map-inference for large scale discrete computer vision problems via combinatorial optimization," in *CVPR*, 2013.

[84] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," *Foundations and Trends® in Machine Learning*, vol. 1, no. 1-2, pp. 1–305, 2008.

[85] D. Tarlow, I. E. Givoni, and R. S. Zemel, "Hop-map: Efficient message passing with high order potentials," in *AISTATS*, 2010.

[86] N. Cornelis, B. Leibe, K. Cornelis, and L. Van Gool, "3d urban scene modeling integrating recognition and reconstruction," *IJCV*, vol. 78, no. 2-3, pp. 121–141, 2008.

[87] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multiview stereopsis," *PAMI*, vol. 32, no. 8, pp. 1362–1376, 2010.

[88] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, *et al.*, "Shapenet: an information-rich 3d model repository," *ArXiv preprint arXiv:1512.03012*, 2015.

[89] *3d warehouse*, `https://3dwarehouse.sketchup.com/`.

[90] K. He, "Mask R-CNN: A perspective on equivariance," in *ICCV Tutorial on Instance-level Visual Recognition*, 2017.

[91] K. Lenc and A. Vedaldi, "Understanding image representations by measuring their equivariance and equivalence," in *CVPR*, 2015.

[92] G. Hinton, *What is wrong with convolutional neural nets?* `https://www.youtube.com/watch?v=Jv1VDdI4vy4`, 2017.

[93] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka, "3d bounding box estimation using deep learning and geometry," in *CVPR*, 2017.

[94] F. Chabot, M. Chaouch, J. Rabarisoa, C. Teulière, and T. Chateau, "Deep MANTA: a coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image," in *CVPR*, 2017.

[95] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun, "Monocular 3d object detection for autonomous driving," in *CVPR*, 2016.

[96] C. Li, M. Z. Zia, Q.-H. Tran, X. Yu, G. D. Hager, and M. Chandraker, "Deep supervision with shape concepts for occlusion-aware 3d object parsing," in *CVPR*, 2017.

[97] P. Poirson, P. Ammirato, C.-Y. Fu, W. Liu, J. Kosecka, and A. C. Berg, "Fast single shot detection and pose estimation," *ArXiv preprint arXiv:1609.05590*, 2016.

[98] F. Massa, R. Marlet, and M. Aubry, "Crafting a multi-task cnn for viewpoint estimation," in *BMVC*, 2016.

[99] S. Tulsiani and J. Malik, "Viewpoints and keypoints," in *CVPR*, 2015.

[100] H. Su, C. R. Qi, Y. Li, and L. J. Guibas, "Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views," in *ICCV*, 2015.

[101] Y. Xiang, W. Choi, Y. Lin, and S. Savarese, "Data-driven 3d voxel patterns for object category recognition," in *CVPR*, 2015.

[102] M. Zia, M. Stark, and K. Schindler, "Towards scene understanding with detailed 3d object representations," *IJCV*, vol. 112, no. 2, pp. 188–203, 2015.

[103] R. Mottaghi, Y. Xiang, and S. Savarese, "A coarse-to-fine model for 3d pose estimation and sub-category recognition," in *CVPR*, 2015.

[104] M. Aubry, D. Maturana, A. Efros, B. Russell, and J. Sivic, "Seeing 3d chairs: Exemplar part-based 2d-3d alignment using a large dataset of cad models," in *CVPR*, 2014.

[105] M. Zia, M. Stark, and K. Schindler, "Are cars just 3d boxes? jointly estimating the 3d shape of multiple objects," in *CVPR*, 2014.

[106] M. Zia, M. Stark, B. Schiele, and K. Schindler, "Detailed 3d representations for object recognition and modeling," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 11, pp. 2608–2623, 2013.

[107] S. Fidler, S. Dickinson, and R. Urtasun, "3d object detection and viewpoint estimation with a deformable 3d cuboid model," in *NIPS*, 2012.

[108] J. Wu, T. Xue, J. J. Lim, Y. Tian, J. B. Tenenbaum, A. Torralba, and W. T. Freeman, "Single image 3d interpreter network," in *ECCV*, 2016.

[109] X. Yan, J. Yang, E. Yumer, Y. Guo, and H. Lee, "Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision," in *NIPS*, 2016.

[110] M. Tatarchenko, A. Dosovitskiy, and T. Brox, "Multi-view 3d models from single images with a convolutional network," in *ECCV*, 2016.

[111] R. Girdhar, D. F. Fouhey, M. Rodriguez, and A. Gupta, "Learning a predictable and generative vector representation for objects," in *ECCV*, 2016.

[112] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum, "Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling," in *NIPS*, 2016.

[113] V. A. Prisacariu and I. Reid, "Shared shape spaces," in *ICCV*, 2011.

[114] ——, "Nonlinear shape manifolds as shape priors in level set segmentation and tracking," in *CVPR*, 2011.

[115] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "Smpl: A skinned multi-person linear model," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 6, p. 248, 2015.

[116] M. M. Loper and M. J. Black, "Opendr: An approximate differentiable renderer," in *ECCV*, Springer, 2014.

[117] D. J. Rezende, S. A. Eslami, S. Mohamed, P. Battaglia, M. Jaderberg, and N. Heess, "Unsupervised learning of 3d structure from images," in *NIPS*, 2016.

[118] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[119] S. Tulsiani, A. Kar, J. Carreira, and J. Malik, "Learning category-specific deformable 3d models for object reconstruction," *PAMI*, 2017.

[120] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[121] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *SIGGRAPH*, 1996.

[122] A. Ghodsi, "Dimensionality Reduction: a short tutorial," University of Waterloo, Ontario, Canada, Tech. Rep., 2006.

[123] K. Li and J. Malik, "Amodal instance segmentation," in *ECCV*, 2016.

[124] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *NIPS*, 2015.

[125] S. Ren, K. He, R. Girshick, X. Zhang, and J. Sun, "Object detection networks on convolutional feature maps," *PAMI*, vol. 39, no. 7, pp. 1476–1481, 2017.

[126] R. Girshick, "Fast R-CNN," in *CVPR*, 2015.

[127] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *ICCV*, 2017.

[128] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[129] B. De Brabandere, X. Jia, T. Tuytelaars, and L. Van Gool, "Dynamic filter networks," in *NIPS*, 2016.

[130] J. C. Spall, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," *IEEE Transactions on Automatic Control*, vol. 37, no. 3, pp. 332–341, 1992.

[131] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: a large-scale hierarchical image database," in *CVPR*, 2009.

[132] G. Varol, J. Romero, X. Martin, N. Mahmood, M. J. Black, I. Laptev, and C. Schmid, "Learning from synthetic humans," in *CVPR*, 2017.

[133] B. Pepik, M. Stark, P. Gehler, and B. Schiele, "Teaching 3d geometry to deformable part models," in *CVPR*, 2012.

[134] Y. Xiang, W. Choi, Y. Lin, and S. Savarese, "Subcategory-aware convolutional neural networks for object proposals and detection," *ArXiv preprint arXiv:1604.04693*, 2016.

[135] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, and R. Urtasun, "3d object proposals for accurate object class detection," in *NIPS*, 2015.

[136] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun, "Monocular 3d object detection for autonomous driving," in *CVPR*, 2016.

[137] D. Forsyth, J. Malik, M. Fleck, H. Greenspan, T. Leung, S. Belongie, C. Carson, and C. Bregler, "Finding pictures of objects in large collections of images," in *Object Representation in Computer Vision II*, vol. 1144, Springer Berlin Heidelberg, 1996, pp. 335–360, ISBN: 978-3-540-61750-1.